Budapesti Műszaki és Gazdaságtudományi Egyetem
Villamosmérnöki és Informatikai Kar
Irányítástechnika és Informatika Tanszék

Tolmácsi Ágnes

DUBINS-JÁRMŰVEK TERÜLETFEDÉSI PROBLÉMÁJA

TDK dolgozat

KONZULENS

Gincsainé Dr Szádeczky-
Kardoss Emese

Tartalomjegyzék

Összefoglaló

Mindig is elgondolkodtatott, hogy egy repülőgép miért párhuzamos egyenesekkel fed le egy területet ahelyett, hogy körkörösen mozogna. Pedig a párhuzamos mozgások során az egyenesek végén meg is kell fordulniuk, amellyel a feladathoz nem tartozó részek felett repülnek el. Ezzel szemben egy porszívó vagy fűnyíró területfedési algoritmusai változatosabbak, nincsenek erre az egyszerű bejárásra korlátozva.

Felmerült bennem a kérdés, hogy ez feltétlenül így kell-e lennie. Tudunk-e egy repülőnek olyan algoritmust mutatni, ahol a megszokott keretekből kilépve költséget tud megtakarítani? Ha differenciális meghajtású robotokra többféle algoritmus alkalmazható, akkor egy Dubins-járműre miért ne lehetne kipróbálni? Sikerülhet-e megugrani, hogy a nagyobb fordulási sugárral rendelkező Dubins-jármű a párhuzamos egyenesek bejárásán kívül is tudjon olyan jó költségű bejárást produkálni, hogy lekörözze a jól bevált módszert? Esetlegesen, ha ez a két más-más tulajdonságokkal rendelkező robot egymást segítve dolgozhatnak, tudnak-e még alacsonyabb költséget felmutatni? A dolgozatomra ezekre a kérdésekre keresem a választ.

A dolgozat Dubins-járművek területlefedési problémájával foglalkozik, ahol a lefedendő terület, valamint a jármű paraméterei adottak és előre ismertek. Ismertetem a lefedéshez alapvető tudnivalókat; Dubins-görbék, a cella-dekompozíció fajtái, a párhuzamos egyenesek módszerének alapjai (boustrophedon út), és a lefedési mutatók.

Egy általam specifikált területlefedő algoritmus kerül bemutatásra, mely során a Dubins-jármű egy „csiga vonalban” indul el a területen, és egy ponton áll át a párhuzamos egyenesek mentén való mozgásra. Ezáltal megspórolja azt a plusz mozgást, amit a területen kívül tenne meg ha párhuzamosokkal fedné le a területet.

Az eredmények szerint van olyan sokszög, amely esetén a módszer jobban teljesít a boustrophedon utaknál, azonban más sokszögeknél rosszabbul. Általánosságban elmondható, hogy a nagyobb szögeket tartalmazó alakzatok esetén az algoritmus kiválóan használható, kis szögeket tartalmazó sokszögek esetén viszont nem alkalmazható hatékonyan.

Ennek megoldására egy kooperációs feladat során a Dubins-jármű mellett egy omnidirekcionális robot is alkalmazható, mely lefedési képességei bár csekélyebbek, viszont mozgása sokkal szabadabb.

A kooperációs feladathoz egy irodalomkutatás történt játékelméleti témában annak érdekében, hogy a kooperációs feladatok során az eredmények számszerűsíthetők és összehasonlíthatók legyenek. Végül a kooperációs megoldás eredményeit értékelem különböző stratégiák esetén.

A megvalósított bejárások implementációja is megtörtént Matlab fejlesztői környezetben. Ezen eredményeket a dolgozat bemutatja, és értékeli.

Abstract

I've always wondered why an aircraft covers an area with parallel lines instead of moving in a circular pattern. After all, during parallel movements, it has to turn at the ends of the straight lines, which involves flying over areas that are not part of the task. In contrast, the algorithms for vacuum cleaners or lawnmowers that cover areas are more diverse and not limited to this simple traversal.

This raised the question for me whether it has to be this way. Can we provide an algorithm to an aircraft where it can save costs by deviating from the usual framework? If differential drive robots can apply multiple algorithms, why not try one for a Dubins vehicle? Can a Dubins vehicle with a larger turning radius find a cost-effective path beyond covering areas with parallel lines, possibly surpassing the well-established method? Perhaps, if these two robots with different characteristics can work together, can they achieve even lower costs? These are the questions I seek to answer in my thesis.

The thesis addresses the area coverage problem of Dubins vehicles, where the area to be covered and the vehicle's parameters are given and known in advance. I provide an overview of essential information for coverage, including Dubins curves, types of cell decomposition, the basics of the boustrophedon method (parallel lines), and coverage metrics.

A coverage algorithm is presented in which the Dubins vehicle starts in a "spiral" path within the area and transitions to moving along parallel lines at a certain point. This saves the additional motion that would occur outside the area if it were to cover it with parallel lines.

The results indicate that there are polygons for which this method outperforms boustrophedon paths, but it performs worse for other polygons. In general, the algorithm works excellently for shapes with larger angles but is not efficiently applicable to shapes with small angles.

To address this, in a cooperative task, an omnidirectional robot can be used alongside the Dubins vehicle. Although its coverage capabilities are lower, its movement is much more flexible.

For the cooperative task, a literature review was conducted in game theory to quantify and compare results in cooperative scenarios. Finally, I evaluate the results of the cooperative solutions under different strategies.

The implementations of the coverage algorithms were carried out in Matlab development environment. The thesis presents and evaluates these results.

1 Bevezetés

A robotok mozgástervezésével számtalan kutatás foglalkozik [1]. A mozgástervezés egyik fajtája az, amikor azt szeretnénk, hogy egy adott területen minden részt bejárjon a jármű, és ezt területfedésnek nevezzük [2]. Robotok területfedését számos helyen alkalmazzák: fűnyírás, takarítás, hókotrás, festés, permetezés vagy szúnyogírtás csak hogy néhányat megemlítsek. Ezeknél a feladatoknál a feladat elvégzése mellett egy másik szempont a robot erőforrásainak minimalizálása. Ilyen erőforrás lehet többek között a mozgáshoz szükséges energia, idő vagy pénzben kifejezett költség.

Ezen erőforrások minimalizálására számtalan algoritmust fejlesztettek ki. Ezek által az egyes mozgások költsége csökkenthető, amelynek nem csupán pénzben kifejezhető előnye van, hanem sokszor ökológiai szempontból képes a bejárást javítani (például CO_2 kibocsátás), ezért fontos az optimalizált algoritmusok keresése.

Dubins-járművek olyan eszközök, amelyek csak előre képesek menni állandó sebességgel és balra vagy jobbra kanyarodni (tehát tolatni, megállni nem tudnak). Ezen járművek területfedése is egy érdekes probléma [3][4]. Itt azzal a nehézséggel kell megküzdeni, hogy a jármű mozgása igen korlátozott, és ezért nagyon fontos, hogy egy előre jól megtervezett útvonalat járjon be a robot. Ez a dolgozat is egy ilyen jármű területfedésével foglalkozik.

A terület adott, a feladat, hogy ezt lefedje a Dubins-jármű. Ez jellemzően lehet egy repülőgép, egy földi kerekeken guruló jármű vagy egy tengeralattjáró, amelyeknek a minimális fordulási sugaruk egy meghatározott érték. Gazdasági szempontból fontos, hogy a mozgás útvonala már az indulás előtt megfelelően meg legyen tervezve, ezzel a lehető legjobban lecsökkentve annak hosszát/idejét. A végső cél a terület legjobb lefedése a költségek minimalizálása mellett.

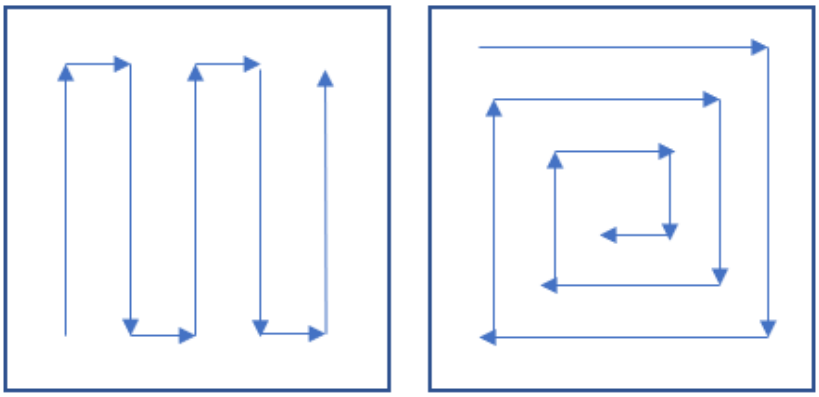
A Dubins-jármű által való lefedés mellett érdekes lehet azt is megvizsgálni, mi történik akkor, ha több robottal próbáljuk meg a problémát megoldani. Ennek megoldására egy kooperációs feladat során a Dubins-jármű mellett egy omnidirekcionálisan mozogni képes robotot is alkalmaztam, mely lefedési képességei bár csekélyebbek, mint egy Dubins-járműé, viszont mozgása sokkal szabadabb, amely segíthet az algoritmus negatívumait kompenzálni. A kooperációs feladatra mérőszámokat

és stratégiákat definiáltam, hogy összehasonlítható legyenek az egyes lefedési módszerek.

A dolgozatban először bemutatom a területfedésben használatos módszereket (2. fejezet), majd részletezem az a Dubins-járművek (3. fejezet) tulajdonságait. A Boustrophedon lefedés bemutatásra kerül (4. fejezet) és ismertetem a CMB algoritmust (5. fejezet). Ez az algoritmus a Circular Motion with Boustrophedon rövidítéséből kapta a nevét, amely egy általam kifejlesztett módszer, és a Dubins-járművek körkörös mozgásával ad lehetőséget a probléma megoldására. A 5. fejezetben bemutatott módszerek implementálásának eredményét az 6. fejezet tartalmazza. A 7. fejezetben a kooperációs területlefedés kerül bemutatásra, majd végül a 8. fejezetben a továbbfejlesztési lehetőségeket tárgyalom.

2 Területfedési útvonal tervezése

A legegyszerűbb területlefedési feladat az, ha egy üres, akadálymentes területet kell bejárni. Két standard megoldás létezik ezen problémákra, ezek az egyenesen oda-vissza, és a spirális mozgás [5]. Ezen mozgások jó alapul szolgálnak bármely valódi rendszerben implementálandó algoritmus számára.



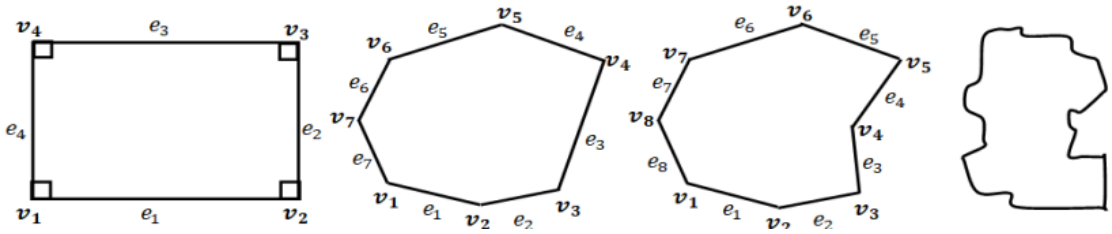
1. ábra Oda-vissza és spirális mozgások

Azonban a legtöbb esetben nem egy egyszerű, akadálymentes terület lefedése a feladat, hanem bonyolultabb, akadályokkal teli. Ebben az esetben cella-dekompozíció alkalmazható, mely során a területet több darabra szétvágva, külön kezelendő részek jönnek létre, amely részeken a területlefedési feladat külön-külön végrehajtható.

Először egy elméleti áttekintés történik a mozgástervezési feladatról, melyben formálisan ismertetem a problémát.

2.1 Területek csoportosítása

A lefedendő területek formája és nagysága igen különbözőek lehetnek. Ha a kétdimenziós megközelítést alkalmazzuk, akkor négy főbb csoportot tudunk elkülöníteni [6]: téglalap, konvex sokszög, konkáv sokszög, nem szabályos alakzat (2. ábra).



2. ábra Téglalap, konvex sokszög, konkáv sokszög, nem szabályos alakzat [6]

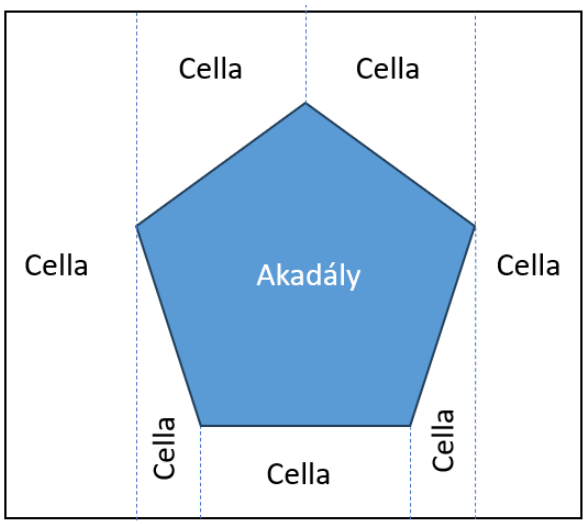
A dolgozatban az első három terület fajtát vizsgálom. Fontos megjegyezni, hogy a negyedik alakzatot sokszor közelítik a harmadik segítségével, mivel jellemzően a területet GPS koordinátákkal adják meg.

2.2 Cella-dekompozíció

A cella-dekompozíció célja, hogy egy összetett, bonyolult problémát (például komplex alakzatot) több egyszerűbbre vezessen vissza azáltal, hogy részegységekre (vagyis cellákra) bontja a területet. A cella-dekompozíció során olyan cellákat kell megalkotni, amik nem metszik egymást, és uniójuk lefedi a szabad, bejárandó területet. Arra ad megoldást, hogy hogyan történjen a terület szétdarabolása. A két legismertebb cella-dekompozíciós módszer a trapéz és a boustrophedon-dekompozíció [7].

2.2.1 Trapéz-dekompozíció

A nevéből is adódóan ez a módszer a területet trapéz részterületekre vágja szét, olyan módon, hogy az akadály csúcsait és a terület szélét párhuzamosokkal összeköti (3. ábra), ezáltal minden celláról azt is meg lehet állapítani, hogy az akadály vagy pedig nem.

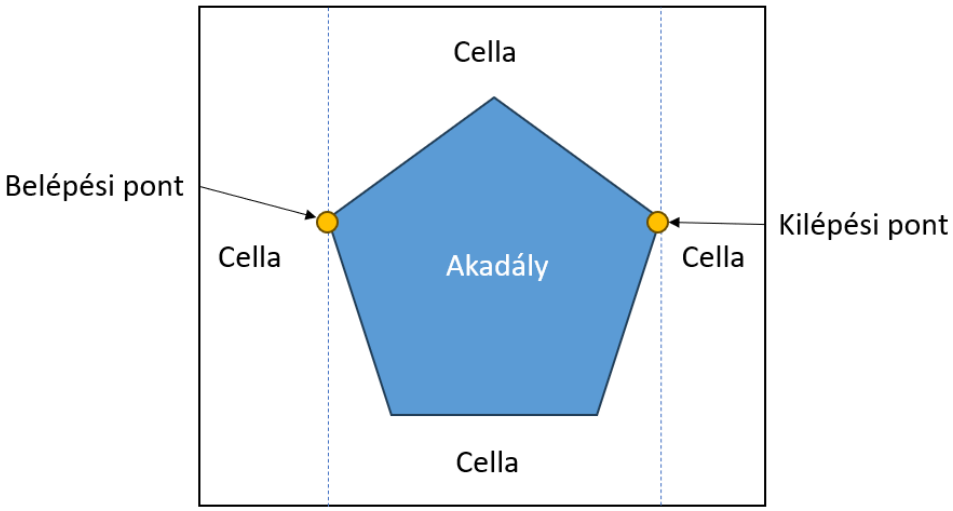


3. ábra Trapéz-dekompozíció

Ennek a módszernek az szab határt, hogy sok kis területet alakít ki, amely szintén problémás lehet a bejárás szempontjából, valamint a valóságban a sokszögek jóval bonyolultabbak lehetnek (példaként tekinthetünk a 2. ábra nem szabályos alakzatára).

2.2.2 Boustrophedon-dekompozíció

Ezen dekompozíciós eljárás a trapéz módszer kiterjesztése olyan módon, hogy összeolvasztja azon cellákat, amelyek az akadály belépési és kilépési pontja között helyezkednek el (4. ábra).



4. ábra Boustrophedon-dekompozíció

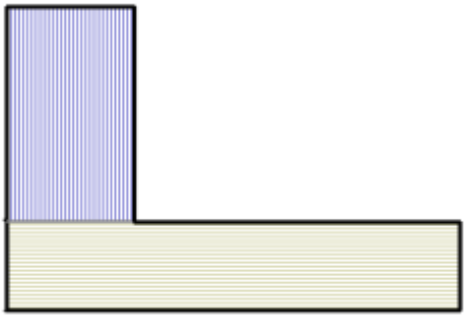
Ebben az esetben a dekompozíció kevesebb részterületet eredményez, viszont ezen területek formája nem egységes, nehezebb bejárhatóságot tesz lehetővé.

A boustrophedon elnevezés eredete, és jelentése a 2.3.1.alfejezetben olvasható.

2.2.3 Akadálymentes terület dekompozíciója

Olyan területek rész egységekre való szétbontására is lehet igény, amelyben akadály nem található, viszont egyszerűbb, könnyebben bejárható alakzatokra osztva a módszerek hatékonyabban alkalmazhatók. Számos szakirodalom foglalkozik ilyen algoritmusok vizsgálatával [8][9][10].

Ezen módszerek igyekeznek a területet minél egyszerűbb alakzatokra szétszedni, és ezeket valamilyen módon bejárni (5. ábra).



5. ábra Rész területek dekopmozícióval [10]

Az 5. ábra bemutatja, hogy a dekompozíció eredményeképp két konvex alakzatot kell lefedni egy konkáv helyett.

2.3 Cellák lefedése

A dekompozíció után (ha egyáltalán a dekompozícióra sor került) következik a részcellák lefedése. A lefedésről elmondható nagy általánosságban, hogy az oda-vissza mozgást használja valamilyen változatban. Ezt másképpen boustrophedon útnak is nevezik [11].

2.3.1 Boustrophedon-út

A boustrophedon egy görög eredetű szó, amelyet először 1699-ben használták, és a jelentése „az ökör iránya” (angolul „the way of the ox”). Tipikusan, amikor egy ökör a földet szántja, azt olyan módon teszi, hogy a terület leghosszabb oldala mentén egy egyenesen végighalad, majd ott visszafordul. Ezt addig folytatja, amíg az egész területet végig nem járta. Az elnevezés tehát innen ered.

Azzal, hogy melyik oldal mentén érdemes a párhuzamos mozgást elvégezni, és mely egyenesről mely egyenesre való áttérés jár a legkevesebb vesztességgel, szintén

több szakirodalom foglalkozik [8][12]. A továbbiakban olyan bejárásokról lesz szó, amelyek a leghosszabb oldal mentén való egyenesekkel dolgoznak. Ennek az az oka, hogy a hosszabb oldal mentén való mozgás (azonos lefedési képességet feltételezve) kevesebb fordulást eredményez.

2.3.2 Egyenesek sorrendje

Az az elv kerül érvényre, miszerint az egyenesek irányától függetlenül a bejárás során a terület lefedéséhez közel ugyanannyi út bejárása szükséges, a módszerek útkülönbségét az egyenesek végén való megfordulás eredményezi. Tehát a cél, hogy minél kevesebb fordulással végezze el a robot a területlefedést.

A tervezést tekintve a legegyszerűbb eset, amikor a robot mindig a szomszédos egyenesre fordul át. Ez akkor lehet jó megoldás, ha a minimális fordulási sugár kisebb (vagy esetleg egyenlő) a befedéshez szükséges egyenesek távolságának felénél. Egyéb esetben sok többlet forgást eredményez. Általánosságban elmondható a járművekről, hogy a minimális fordulási sugár kétszerese nagyobb, mint amennyit az egyik egyenesről a másikra fordulni kell.

A bejárás történhet úgy is, hogy nem szomszédos egyenesek között vált a jármű, hanem a lefedési képességet és a minimális fordulási sugarat figyelembe véve megállapítja, hogy hány egyenesenként érdemes kanyarodni. Ezt a számítást az alábbi módon lehet megtenni, ahol az n azt jelöli, hogy hány egyenesenként kell végig menni, az R_{min} a robot minimális fordulási sugara, a w pedig a jármű lefedési képessége ($\lceil kifejezés \rceil$ a felső egészrészt jelenti):

(2.1)

$$n = \left\lceil \frac{2 \cdot R_{min}}{w} \right\rceil$$

Látható, hogy igen sok féleképpen be tudjuk járni az egyeneseket, nevezetesen $(m-1)!$ darabszámú bejárás lehetséges (ahol az m az egyenesek száma), amennyiben az első egyenes adott azt nem tudjuk megválasztani, valamint a végpont nincs előírva. A lehetséges bejárások közül nem mind releváns a minimális útkeresés szempontjából, de azt próbáltam szemléltetni, hogy olyan sok lehetőség van, hogy nem lehet tudni előre, hogy melyik bejárás eredményezi a minimális úthosszt.

3 Dubins-járművek mozgása

A Dubins járművek tulajdonsága, hogy csak egy irányba állandó sebességgel képesek közlekedni v sebességgel (tehát tolatni, megállni, lassítani nem), valamint rendelkeznek egy R_{min} minimális fordulási sugárral. Ilyen járművek lehetek például tengeralattjárók, repülőgépek vagy speciális földön mozgó robotok. Definiálhatunk nekik egy w lefedési képességet jellemző mutatót, amely azt adja meg, hogy ha egy terület felett elhalad, mekkora területet fedett le vele valójában (jellemzően a méter a mértékegysége).

3.1 Dubins-járművek mozgásegyenlete

Elsőként tekintsük meg a differenciális robotok mozgásegyenletei (6. ábra):

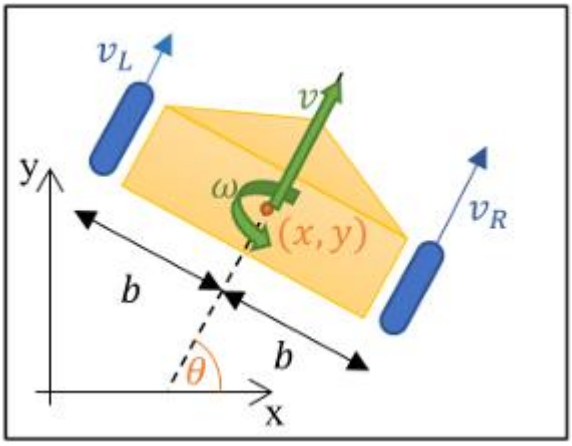
(3.1)

$$\dot{x} = v \cdot \cos\theta$$

$$\dot{y} = v \cdot \sin\theta$$

$$\dot{\theta} = \omega$$

A v a robot sebessége, a θ az orientációja (tengelyének bezárt szöge az x tengellyel), az (x, y) pedig a jármű koordinátája.



6. ábra Differenciális robot modellje

Nézzük hogyan változik mindez a Dubins-járművek esetében. A Dubin-járművek sebessége konstans, tehát a robot mozgásegyenletében a $v = 1$ egyszerűsítéssel élhetünk. Ezen kívül az orientáció változás csak attól függ, hogy jobbra vagy balra kanyarodik, amelyet az u értéke fog megmondani. Ha $u = 0$ akkor a robot előre mozog, ha $u = 1$,

akkor jobbra, ha pedig $u = -1$, akkor balra fordul (7. ábra). Az u lehet 1-től különböző értékű, akkor egy R_{min}-nél nagyobb sugarú körön tud mozogni a robot. Ez viszont nem optimális, ezért jellemzően csak a ± 1-et használjuk.

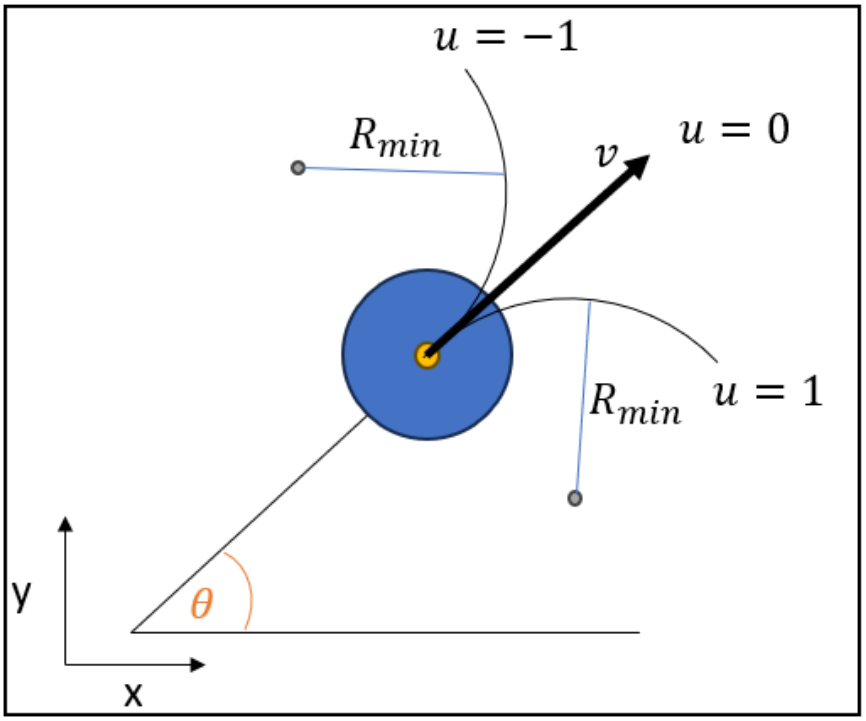
Ezek alapján a mozgásegyenletek az alábbiak szerint alakulnak:

(3.2)

$$\dot{x} = \cos\theta$$

$$\dot{y} = \sin\theta$$

$$\dot{\theta} = u$$



7. ábra Dubins-jármű modellje

Ezen a kinematikai modellek kis sebességre alkalmazhatók, nagyobb sebességnél dinamikus modellt szükséges alkalmazni.

3.2 Dubins-út

Minden Dubins út három mozgásprimitívből áll, amely mozgásprimitívek az S (straigth, egyenes), L (left, bal) és az R (right, jobb) [13]. Ezen kívül fontos, hogy az optimális pálya maximális jobbra és balra kanyarodással érhető el. Ha megnézzük hány féle fordulás lehetséges ezen három primitív összerakásából, akkor kiderül, hogy

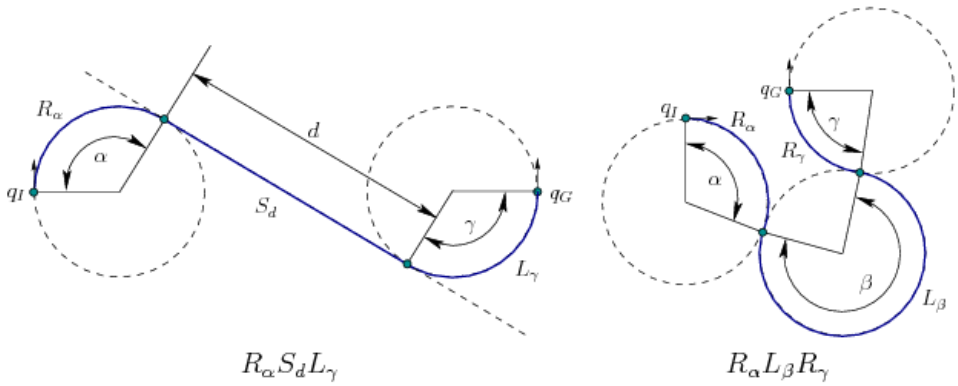
összesen tizenkettő darab (nem követheti egymást két egyforma mozgás, azok összevonódnak):

$$\{SLR, SRL, SRS, SLS, RLS, RSL, RLR, RSR, LRS, LSR, LSL, LRL\}$$

Lester Eli Dubins matematikus, akinek a nevéhez fűződik ezen módszer kidolgozása bebizonyította, hogy összesen hat kombináció eredményezheti a minimális utat:

$$\{L_\alpha R_\beta L_\gamma, R_\alpha L_\beta R_\gamma, L_\alpha S_d L_\gamma, L_\alpha S_d R_\gamma, R_\alpha S_d L_\gamma, R_\alpha S_d R_\gamma\}$$

Az $\alpha \in [0,2\pi)$ megadja, hogy hány fokos ívet fut be a jármű a minimális fordulási sugarú körön az első szakaszban ($\beta \in (\pi, 2\pi)$ és $\gamma \in [0,2\pi)$ hasonlóan rendre a második és harmadik szakaszon), $d > 0$ pedig az egyenes szakasz hosszát jelöli. Az alábbi ábrán (8. ábra) látható egy példa az $R_\alpha S_\beta L_\gamma$ és az $R_\alpha L_\beta R_\gamma$ esetekre.



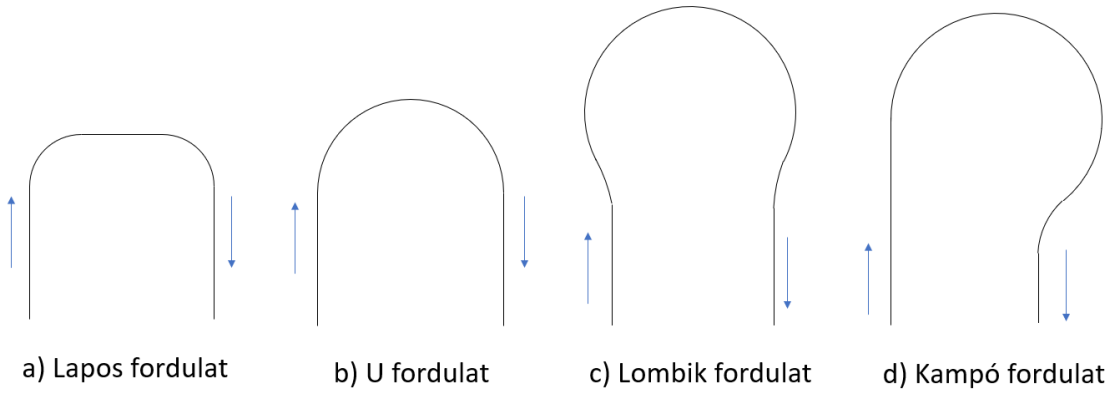
8. ábra Dubins-út példa [13]

Tehát területfedéskor az egyik egyenes végpontjáról a másik egyenes kezdőpontjáig történő mozgás során ezen hat lehetőség egyikét érdemes alkalmazni. A hat közül a legmegfelelőbbet sejteni lehet, azonban meghatározni vagy az összes kipróbálásával és összehasonlításával (brute force módszer), vagy pedig bonyolultabb irányításelméleti eszközökkel lehet.

3.3 Lehetséges átfordulások

Egy Dubins jármű átfordulásai a fent említett hat lehetséges Dubins-útból fog kikerülni, vegyük sorra azon átfordulásokat, amely két párhuzamos egyenes között az átmenetelt tudja biztosítani.

Ezen átfordulásokat négy csoportba sorolhatjuk [10]; ezek az lapos fordulat (flat turn), ’U-fordulat’, lombik alakú fordulat (bulb turn) és kampó fordulat (hook turn) (9. ábra). .



9. ábra Fordulatok bemutatása

Ha ezen fordulatokat Dubins szekvenciával szeretnénk kifejezni, akkor a lapos fordulat a $R_\alpha S_d R_\gamma$, az lombik fordulat a $L_\alpha R_\beta L_\gamma$, a kampó fordulat pedig a $R_\alpha L_\beta R_\gamma$ szekvenciának felel meg. Az U fordulatot egy egyszerűbb, de speciális eset, ezt a R_π primitívvel lehet kifejezni.

4 Boustrophedon lefedés szimulációja

A 2.3.1. alfejezetben bemutatott boustrophedon-utakat implementáltam azon célból, hogy a különböző bejárások hatékonysága vizsgálható lehessen, valamint a későbbiekben összehasonlítás alapjául szolgálhasson. A leírt két módszer a szomszédos egyenesekre való áttérés, valamint a minimális fordulási sugár és a bejárandó egyenesek távolsága alapján meghatározott következő egyenes módszer. Két terület esetén vizsgálom mind kettőt (a későbbiekben fontos lesz egy konkáv eset bemutatása, azonban a párhuzamos egyenesek mentén való bejárás esetén nincs lényegi különbség, emiatt ebben a fejezetben nincs külön tárgyalva).

4.1 Általános és a szimulációban vizsgált mérőszámok

Az algoritmusok használhatóságának vizsgálatához, valamint ahhoz, hogy a különböző tulajdonságaikat össze lehessen hasonlítani, definiálunk hatékonysági mutatókat [6]. Ezen hatékonysági mutatók számszerűen megadják, hogy egy bejárás az adott mutató szerint mennyire helytálló.

Az elsődleges szempont nagy általánosságban a teljes lefedettség. Ebből kifolyólag fontos mérőszámnak számít. Képlettel kifejezve (C: lefedettség, $T_{lefedett}$: lefedett terület, T_{teljes}: teljes terület):

$$C = \frac{T_{lefedett}}{T_{teljes}}$$

Megadja, hogy egy lefedendő terület hány százalékát járta be az algoritmus segítségével a robot.

Ezen kívül fontos szempont a költségek minimalizálása (L), tehát az energia (permetező-, tisztítószer, benzin, energia) felhasználás optimuma is.

A mozgás időtartama (T), valamint a bejárt út (s) által kerül az energiafelhasználás minimalizálva. Minden lefedő algoritmus célja, hogy minél gyorsabban, minél rövidebb idő alatt, minél kevesebb út megtételével elvégezze a feladatot.

Az út hossza kiszámolásra került minden implementált algoritmus esetén. A bejáráshoz szükséges idő a sebesség ismeretében már könnyen számolható. Az

üzemanyag minimalizálását a kooperációs fedés esetében részletesen taglalom, az egy eszközzel való lefedés során az állandó sebesség feltételezése mellett, az energia felhasználás a bejárt úttal egyenesen arányos.

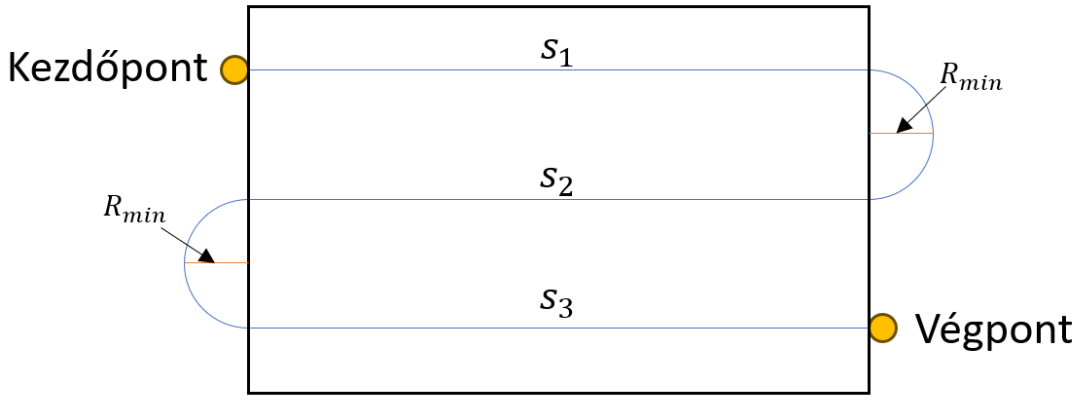
A párhuzamos egyenesek mentén való bejárások esetén nehéz a minimális úthosszt eredményező bejárást megtalálni, ezért az új megközelítésben használt algoritmust a párhuzamos egyenesek módszerének alsó becsléséhez is hasonlítom összevetés céljából.

Az alsó becslés (s_{min}) meghatározásához szükség van minden egyenes hosszára (s_m), az egyenesek számára (m), valamint a minimális fordulási sugárra (R_{min}).

(4.1)

$$s_{min} \leq (m-1) \cdot \pi \cdot R_{min} + \sum_{1}^{m} s_m$$

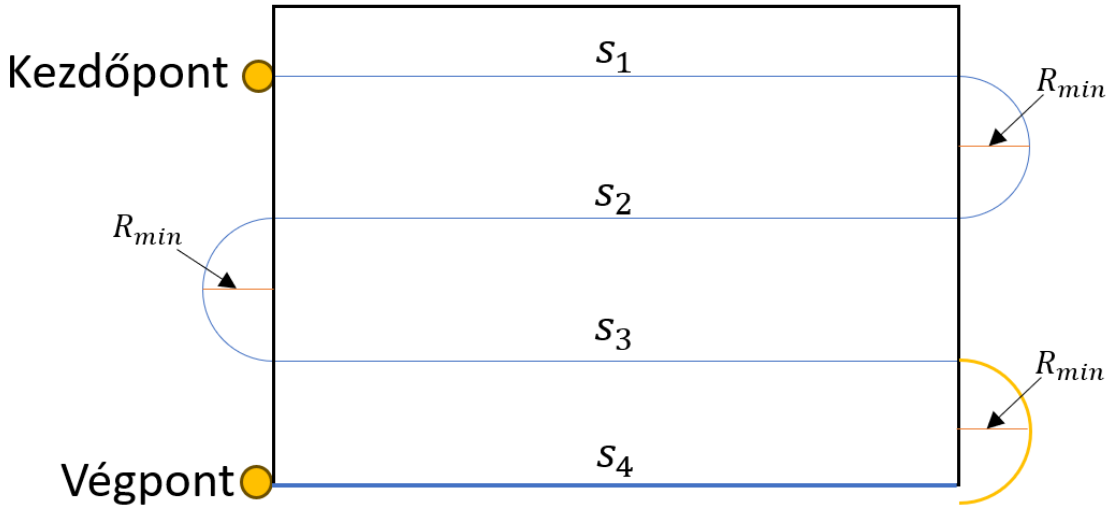
Ez az alsó becslés azért használható, mert az egyenesek adottak, azokon biztosan végi kell mennie a járműnek, valamint az utolsó egyenesen kívül minden egyenes végén fordulnia kell, amely legjobb esetben is egy R_{min} sugarú félkör. Megértést segíti a 10. ábra (ebben az esetben $m = 3$).



10. ábra Alsó becslés ábrázolás: ideális eset

A 11. ábra egy olyan esetet mutat, hogy nem minden egyenes van $2 \cdot R_{min}$ távolságra egymástól ($m = 4$). Ebben az esetben az utolsó átfordulást nem tudja megcsinálni a jármű egy U fordulattal, egy lombik vagy kampó fordulatra lenne szükség. A lombik és kampó fordulat útköltsége viszont azonos R_{min} esetén biztosan nagyobb

lesz, mint egy U fordulaté. Ezért mondhatjuk azt, hogy a (4.1)-es egyenlet a bejárásnak egy alsó becslését adja.



11. ábra Alsó becslés ábrázolása: nem ideális eset

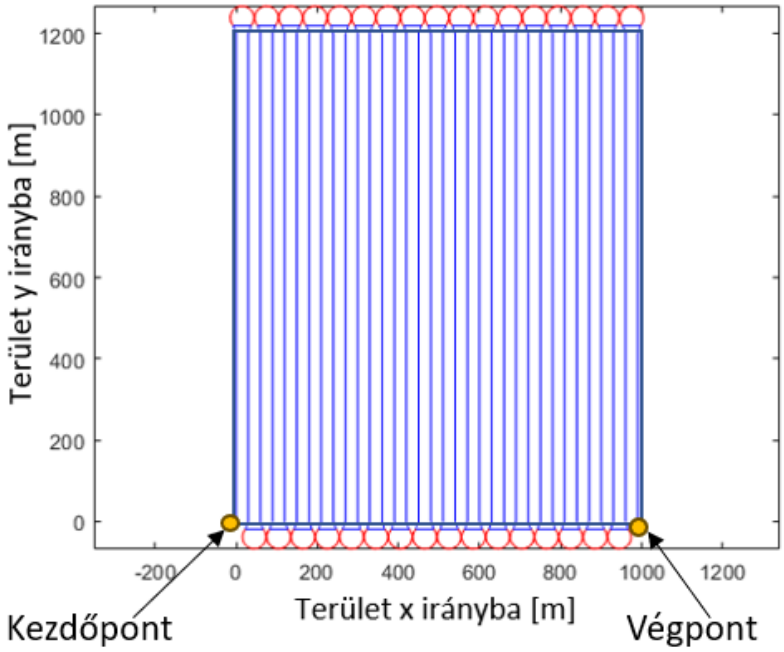
A lefedettség közel 100% minden vizsgált esetben, ennek a mérőszámnak a pontos meghatározására a dolgozat nem terjed ki.

4.2 Implementált bejárások

A bejárás, ahol lehet a leghosszabb oldal mentén történik [14], hogy a fordulások számát minimalizáljuk. Az egyenes végén a megfordulás pedig lombik alakú vagy lapos fordulat. Ha a téglalap olyan tulajdonságokkal bír, hogy a rövidebb oldala nem osztható *w*-vel (lefedési képesség), akkor az utolsó egyenesen a jármű már olyan területeket is lefed, amit már egyszer lefedett.

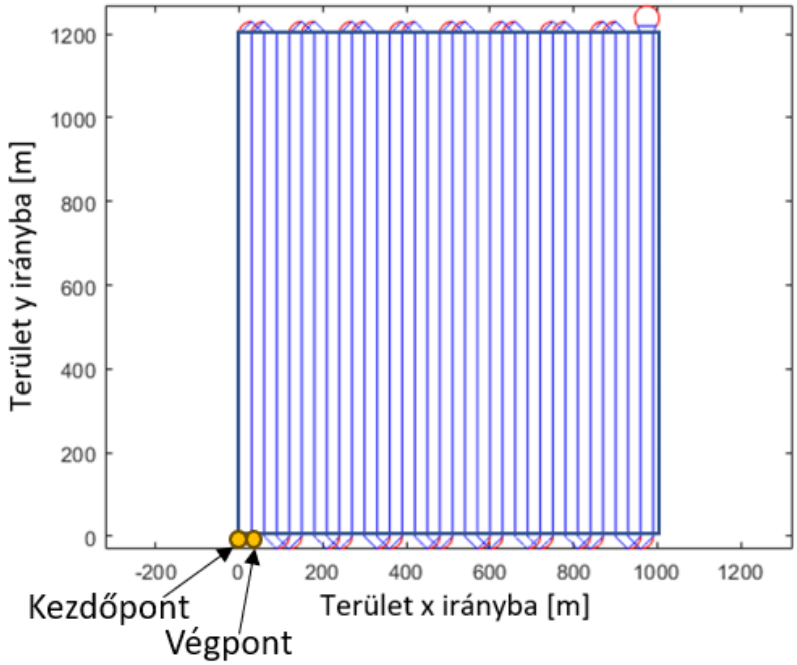
Az első terület egy téglalap az egyik legegyszerűbb munkatérnek tekinthető. Megfigyelhető, hogy az egyenesek irányának kiválasztása a szerint működik, hogy melyik a leghosszabb oldal.

A 12. ábra azt a szimulációt mutatja, amikor a jármű mindig a szomszédos egyenesre tér át.



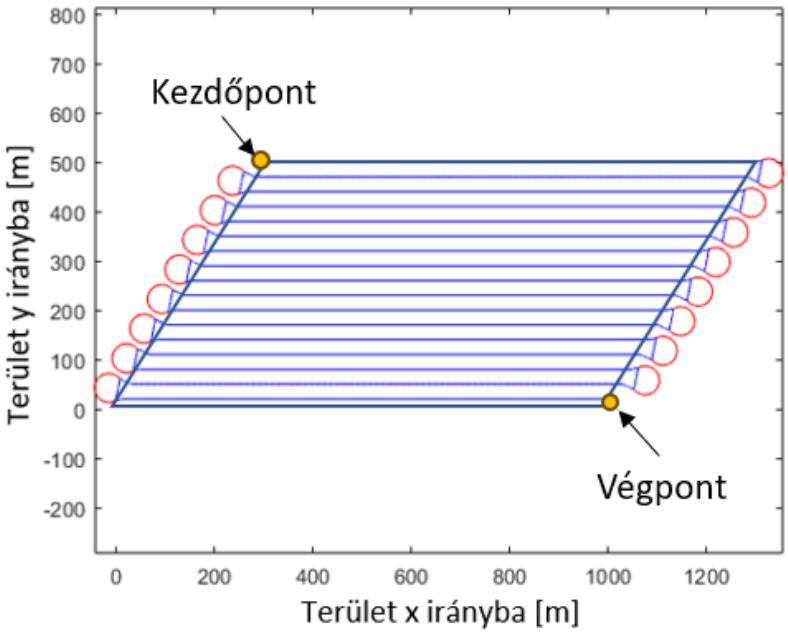
12. ábra Téglalap szomszédos boustrophedon-út

A 13. ábra pedig az optimalizált bejárást mutatja, ahol a (2.1)-es egyenlet alapján számolható ki, hogy hány egyenesenként megy végig a robot a területen. Jelen esetben ez a szám 2-nek adódott.

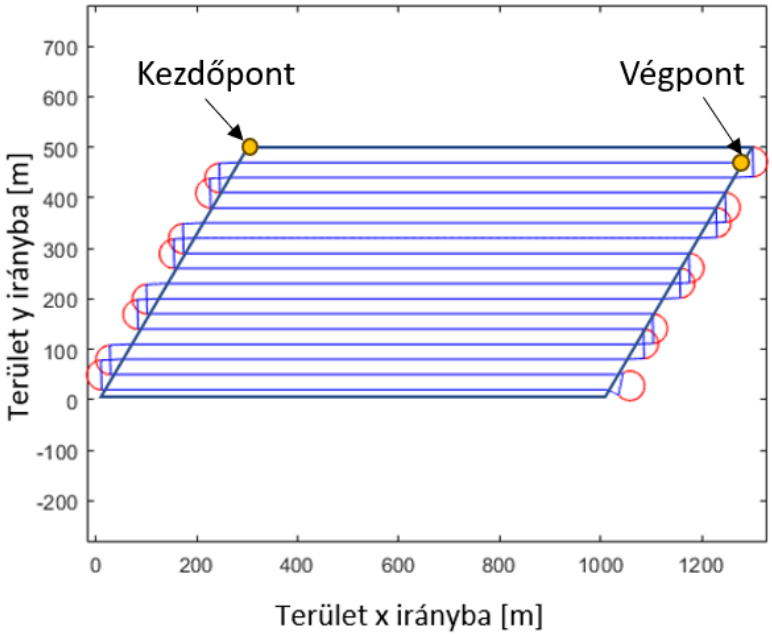


13. ábra Téglalap optimalizált boustrophedon-út

A második területként egy paralelogrammát tekintsünk. Az előzőekhez hasonlóan a 14. ábra a szomszédos áttérés, a 15. ábra pedig az optimalizált áttérés szimulációs eredményét mutatja.



14. ábra Paralelogramma szomszédos boustrophedon-út



15. ábra Paralelogramma optimalizált boustrophedon-út

A hatékonysági mutatók a következők (a mértékegység méter):

	Téglalap	**Paralelogramma**
Szomszédos (1.) módszer hossza [m]	46 530	19 731
Optimalizált (2.) módszer hossza [m]	43 896	18 548
Alsó becslés hossza [m]	43 730	18 373
1./2. módszer	1.06	1.068

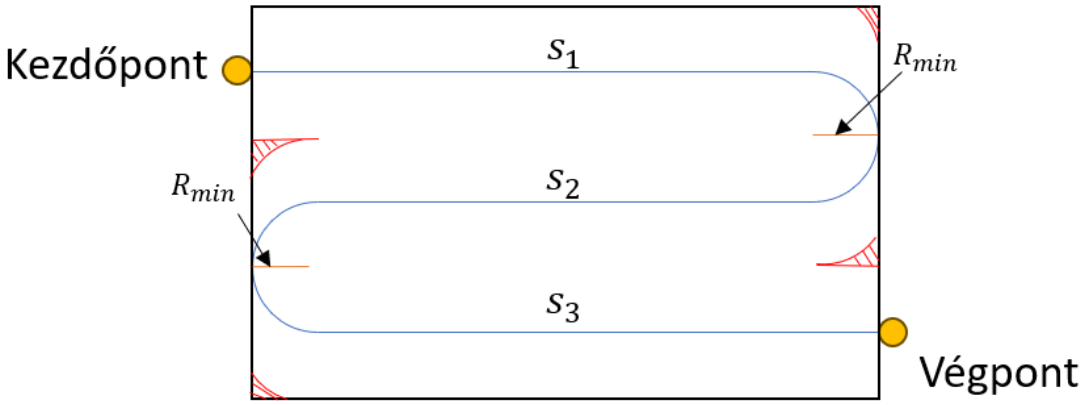
1. táblázat Boustrophedon-utak összehasonlítás

Az 1. táblázat adatai alapján, a 2. módszer a téglalap esetén 6%-kal, a paralelogramma esetén pedig 6.38%-kal bizonyult jobbnak. Mindkét esetben az optimalizált módszer megközelíti az alsó becslést.

A két módszer közötti eltérés nagyban függ a Dubins-jármű adataitól. Könnyen belátható, hogy ha a minimális fordulási sugár kétszerese megegyezik (vagy kisebb), az a lefedési képességet jellemző távolsággal, akkor a két módszer ugyanazt az eredményt adja.

5 Circular Motion with Boustrophedon (CMB) algoritmus

A boustrophedon-utak nagy hátránya, hogy az alakzaton kívül is végez a jármű mozgást, amely nem számít effektív útnak, másrészről nem lehet feltétlen tudni, hogy a sokszögön kívül hol van megengedett vagy tiltott zóna. Ezen második problémát ki lehetne küszöbölni egy „alakzat kiterjesztéssel", amely esetben azt a területet is számításba vesszük, ahol a robot a megfordulásokat végzi, viszont a felesleges utak problémája ezzel megoldatlan maradna. Ha a párhuzamos egyenesek módszerével a téglalapon belül történne a megfordulás, akkor a lefedettség mérőszáma sérülhetne (16. ábra). A piros területek mutatják azokat a területeket, amelyeket a jármű a területen belüli megfordulással nem fed le.



16. ábra Területen belüli megfordulás

5.1 Általánosan a CMB-ről

Az alap ötletem az volt, hogy bármilyen forgás is történik, az legyen az alakzaton belül mindez úgy, hogy törekedni kell a többszörös lefedések minimalizálására. Ezért a Dubins-jármű egy csigaházra hasonlító mozgással indul el az alakzatban, majd később tér át a párhuzamos egyenesek módszerére.

A csigavonal diszkrét pontokkal van definiálva, mely pontokat a járműnek érintenie kell az útja során (kivéve a csúcspontokat). Az eredeti alakzat oldalaival párhuzamosan mozog, és az egyik oldalról a másikra tér át. Az áttérés módja a jármű és a bejárandó terület paramétereitől is függ. Ezt így folytatja, amíg a minimális fordulási

sugarának kétszerese összemérhetővé válik az egyre kisebb sokszög oldalaival, ilyenkor áttér a párhuzamos egyenesek módszerére. A pontos lépéseket a következőkben részletezem.

5.2 Lépések

Az 5.2.1, 5.2.2 és a 5.2.3 pontok az alakzat feltérképezésére szolgálnak, ezek tetszőlegesen felcserélhetők.

5.2.1 Kezdőpont megkeresése

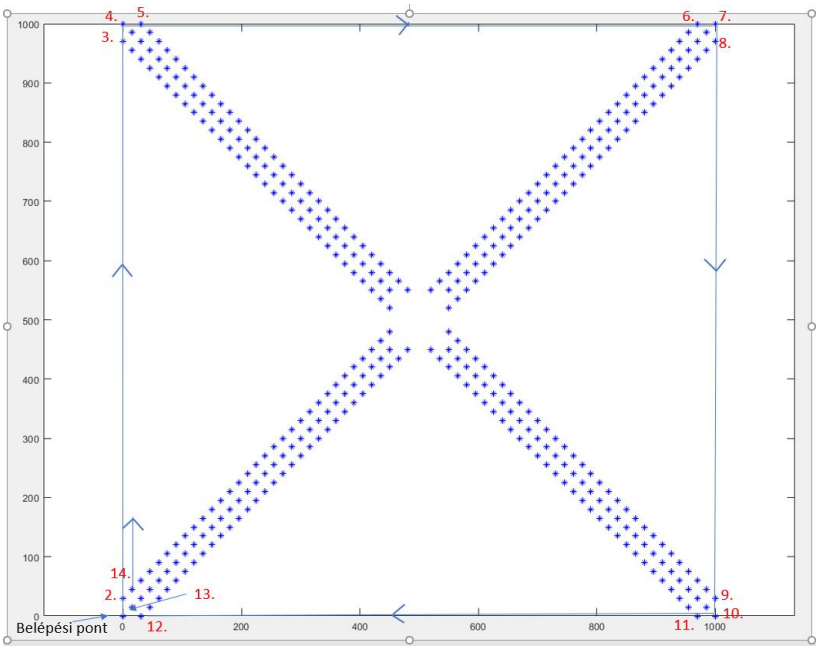
A kezdőpontnak igen egyszerűen azt a csúcspont lesz, amely a sokszög legkisebb szögéhez tartozik. Ennek az az oka, hogy kisebb szögeket nehezebb a járműnek bevennie, és a későbbiekben az alakzaton belüli forgásoknál így eggyel kevesebb lesz a legkisebb szögből. Amennyiben több azonosan kicsi szög is van, a kezdőpont ezek halmazából tetszőlegesen választható.

5.2.2 Bejárandó pontok meghatározása

A kezdőponttól függetlenül meghatározható a bejárandó pontok halmaza, viszont a bejárás sorrendje a kezdőpont meghatározása után derül ki.

A sokszög oldalai mentén minden csúcstól adott távolságra (18. ábra jelölése szerint x) összesen két pontot jelölünk ki. Ezután a sokszöget kicsinyítjük olyan módon, hogy a kicsinyített alakzat hasonló legyen az eredetihez, az oldalai párhuzamosak legyenek az eredeti oldalakkal, valamint w (lefedési képesség) távolságra legyenek egymástól. Van olyan eset, hogy ez nem teljesíthető, mert olyan kicsi egy oldal, hogy a kicsinyítések közben eltűnik, ilyenkor két csúcs összeolvad.

Addig folytatódik a kicsinyítés, amíg a legkisebb oldal kisebb nem lesz, mint négyszer a minimális fordulási sugár. Ennek az oka, hogy ha még egy iterációt mennénk, akkor már a legrosszabb esetben kevesebb, mint kétszer R_{min} távolságra lennének a pontok, amelyet a jármű egy ívben biztos, hogy nem tud bevenni. Egy négyzetet tekintve a 17. ábra mutatja a generált pontokat.



17. ábra Generált pontok

5.2.3 Minimális bevehető szög meghatározása

Amikor a Dubins-jármű egy csúcshoz ér, ebbe beleértve a kicsinyített sokszögek csúcspontjait is (a belépési pontnál a csúcspontok el vannak csúszva kanyarodás szempontjából), el kell döntenie, hogy a kanyart milyen módon képes bevenni. Erre két lehetősége van.

Az első az út szempontjából előnyösebb az $R_\alpha S_d R_\gamma$ szekvenciájú Dubins-utas áttérés (lapos fordulat), mivel óramutató járásával megegyezően mozog a robot, és így ez a kedvezőbb azzal a kiegészítéssel, hogy határesetben az S szakasz eltűnhet, és az átfordulás egy R_{min} sugarú kör mentén történhet.

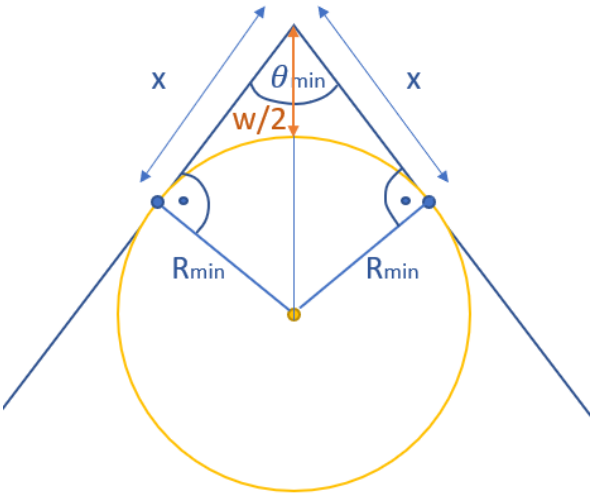
A második esetben, amikor nem tud az első esetben felvázolt módon áttérni, választhatja az $L_\alpha R_\beta L_\gamma$ szekvenciát (lehet, hogy más szekvencia lenne a legrövidebb pályát eredményező, ez csak egy lehetséges áttérés). Ez egy jóval hosszabb utat adó megoldás, mint az első lehetőség.

Ahhoz, hogy a robot dönteni tudjon a két vázolt kanyarodás közül kiszámolandó az a határszög, amelynél még képes az első kanyarodást végrehajtani.

(5.1)

$$\theta_{min} = 2 \cdot \operatorname{asin}\left(\frac{R_{min}}{\frac{w}{2} + R_{min}}\right)$$

Az w a lefedési képességet jellemző távolság, és az R_{min} a minimális fordulási sugár. A képlet a 18. ábra alapján határozható meg, egyszerű geometriai megfontolások alapján. Megállapítható, hogy a w növekedésével csökken a minimális szög.

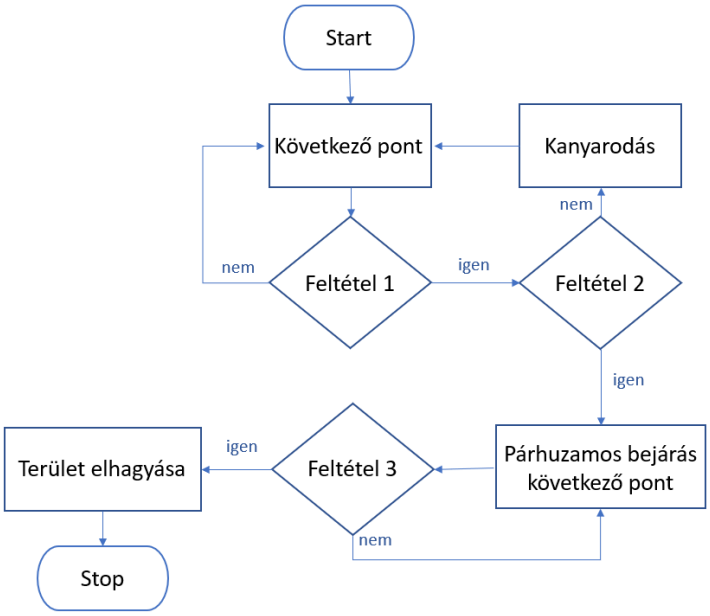


18. ábra Minimálisan bevehető szög

5.2.4 Pontok bejárása

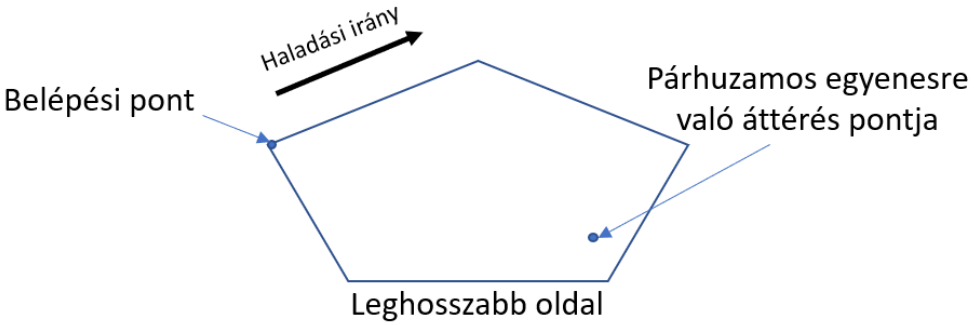
A pontok bejárásához szükséges input a pontok sorszámokkal ellátott halmaza (vagyis az 5.2.1 és a 5.2.2 alfejezetek kimenete).

A bejárás folyamatdiagramját a 19. ábra mutatja be. A ’Feltétel 1’ azt vizsgálja, hogy a pont csúcspont-e, a ’Feltétel 2’ azt, hogy a párhuzamos bejárásra való átállás esedékes-e, a ’Feltétel 3’ pedig, hogy bejárta-e az egész területet.



19. ábra Algoritmus folyamatdiagram

A robot belép az alakzatba (az első ponton áll). Ezek után megnézi a következő pontot. Ha nem csúcspont, akkor mozog a következő pontra, ha igen, akkor megvizsgálja, hogy a párhuzamos mozgásra való átállásnak meg kell-e történnie. A jármű alap esetben a legkisebb sokszögben megy a párhuzamos egyenesek mentén viszont, ha nem ott fejezi be a körkörös mozgást, ahol a sokszög leghosszabb oldala kezdődik, akkor az eggyel nagyobb sokszög megfelelő csúcsánál vált. Ez azért történik így, mivel a leghosszabb oldal mentén érdemes a párhuzamos egyeneseket húzni. Ennek megértését segíti a 20. ábra.



20. ábra 'Feltétel 2' magyarázata

A belépési pont a legkisebb szögnél van, viszont a leghosszabb oldal kezdőpontja ezzel nem esik egybe. Tehát a párhuzamos egyenesekre való áttérés nem valósulhat meg

a legkisebb sokszögben, ebben az esetben például a legkisebb sokszög első pontja előtt két ponttal megtörténik a váltás.

Az áttéréskor a megmaradt pontokból egy „új sokszöget” tekintünk, ahol az egyik boustrophedon-utas módszer végigfut. Ha a belépési pontnál kezdődik a leghosszabb oldal (óramutató járásával megegyezően nézve), akkor az új sokszög a legkisebb sokszöggel megegyezik. Azonban, ha ez nem teljesül, akkor ettől eltérő a sokszög (a belépési pont kicsinyített pontjáig az eggyel nagyobb sokszög pontjai, onnantól a legkisebb sokszögé).

Végül az algoritmus megvizsgálja, hogy az egész kis sokszöget bejárta-e, ha igen akkor a bejárás befejeződik.

5.3 Előnyök, hátrányok

A módszer egyik nagy előnye, hogy nem hagyja el a jármű a területet (kivéve, ha van a minimálisan bevehető szögnél kisebb a sokszögben). Egyrészt igyekszik ezzel minél kevesebb felesleges utat megtenni, másrészről nem érint olyan területet, melynek a státusza ismeretlen.

Másik előnye, hogy azon alakzatok esetén, ahol nincs olyan szög, amely nem éri el a minimálisan bevehető szöget, a kipróbált esetekben a párhuzamos egyenesek módszerének elvi minimumánál kevesebb utat tesz meg.

Hátránya, hogy többszörös lefedések vannak az alakzatban, nehezebb megállapítani a lefedettséget. Ezen kívül, ha a sokszög tartalmaz olyan szöget, amely nem éri el a minimálisan bevehető szöget, akkor az útvonal több lesz, mintha párhuzamos egyenesekkel haladna végig a terület felett.

Az, hogy hatékony-e a módszer a sokszög és a Dubins-jármű paraméterei alapján egyértelműen megállapítható, a bejárás előtt el lehet dönteni, hogy megéri-e ezt használni ((5.1)-es egyenlet).

6 CMB algoritmus szimulációs eredmények

Különböző sokszögekre szimuláltam a 5. fejezetben bemutatott CMB algoritmust. Ez néhány példán keresztül kerül bemutatásra, valamint mérőszámokkal összehasonlítom az egyes sokszögekre az algoritmust az eddig bemutatottakkal.

Mivel a boustrophedon-utak vizsgálatánál az lett az eredmény, hogy az optimalizált párhuzamos egyenes bejárás a kedvezőbb, így ezen algoritmus esetén is ez alkalmazandó.

6.1 Téglalap bejárása CMB-vel

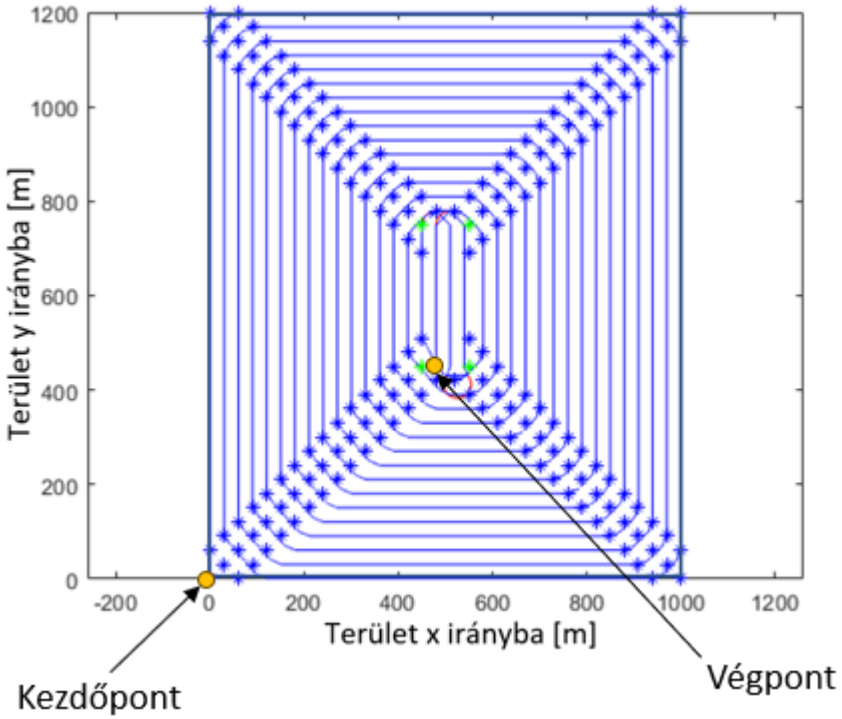
Az első példa egy olyan alakzat, ahol az alakzat minden szöge a minimálisan bevehető szögnél nagyobb. A szimulációhoz használt értékek (a szimuláció paraméterei egy repülőgépet vesznek alapul, az R_{min} értéke függ a jármű sebességétől, az $\alpha_{\min\ \_turn}$ a (5.1)-es egyenlet alapján pedig a w és az R_{min} értékektől):

$$R_{min} = 29.144\ m$$

$$w = 30\ m$$

$$\alpha_{\min\ \_turn} = 82.63°$$

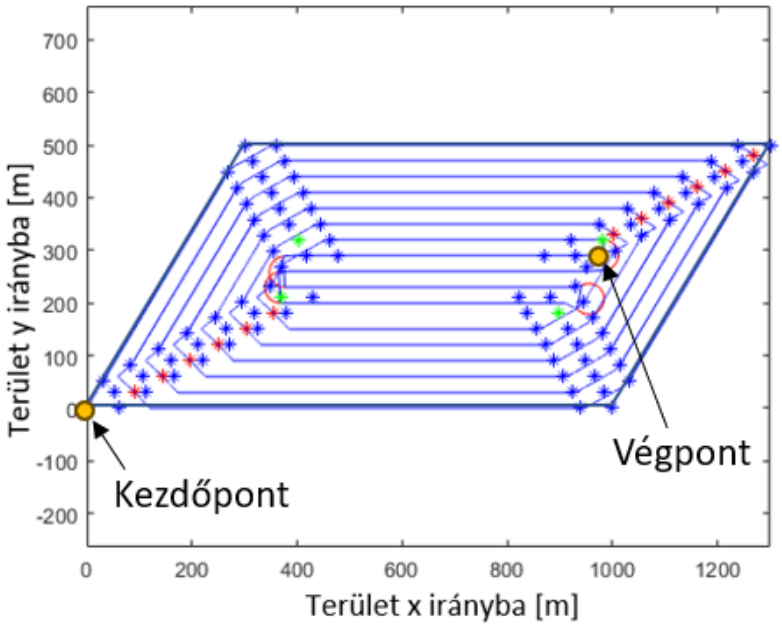
Az ábrán láthatók a generált (kék) pontok, valamint a bejárási út. A négy zöld pont pedig, az új kis terület, amelyet párhuzamos egyenesekkel jár be a Dubins-jármű.



21. ábra Téglalap bejárása

6.2 Paralelogramma („kis szögű” sokszög)

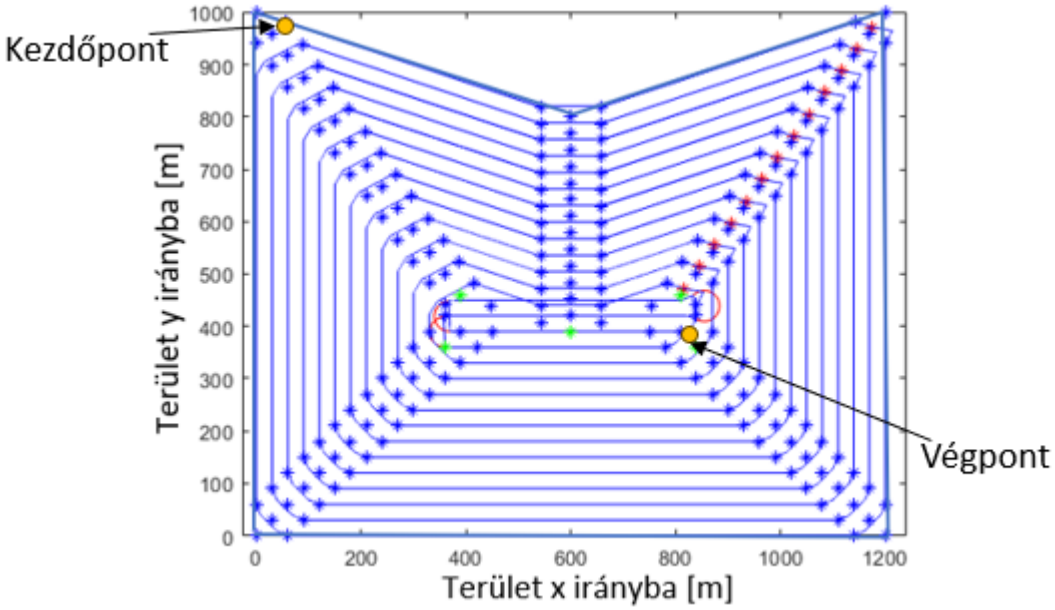
Ebben a példában az alakzat tartalmaz olyan szöget, amely nem éri el a minimálisan bevehető szöghatárt. Például egy „lapos” paralelogramma. A paralelogramma hegyesebb szögeinél (22. ábra piros pontok) a megfordulás egy hosszabb utat eredményez, mivel a $L_\alpha R_\beta L_\gamma$ szekvenciát alkalmazza a robot (a 22. ábra ezen fordulásokat nem mutatja). A paralelogramma hegyesebb szöge jelen esetben $59°$ a tompaszöge pedig $121°$.



22. ábra Egy paralelogramma bejárása

6.3 Konkáv alakzat

Konkáv esetben az a különbség adódik, hogy a homorú szögnél a jármű a külső íven mozog nem pedig a belsőn, mint a többi csúcsnál. Itt is fennáll az az eshetőség, hogy a homorú szög kiegészítő szöge kisebb lehet, mint a minimálisan bevehető szög.



23**. ábra Konkáv alakzat bejárása**

A konkáv alakzat hegyesszögei a példában 71.57°, a homorú szöge pedig 216.86°.

Ebben az esetben jól megfigyelhető a kezdő pont választásának a fontossága, mivel a bal felső sarokból való kiindulás következtében a további iterációkban a jármű be tudta venni azt a szöget plusz forgás nélkül, míg az azonos nagyágú szöget (a jobb oldali csúcsnál) nem. A kezdőpontnál a robot minden körben eggyel beljebb lép, és emiatt a bevenni kívánt szög nagyobb lesz. Itt a paraméterek pont úgy adódtak, hogy az alapszöget még nem, de a módosított szöget már plusz forgás nélkül teljesíteni tudja.

6.4 CMB és boustrophedon összehasonlítás

A 2. táblázatban feltűntetett értékek egy-egy alakzat bejárához szükséges utakat mutatják méterben kifejezve.

	Téglalap	**Paralelogramma**	**Konkáv sokszög**
CMB hossza [m]	40 283	22 355	45 153
Boustrophedon alsó becslés (B) hossza [m]	43 730	18 373	43 730
CMB/B	0.921	1.217	1.033

2. táblázat CMB-B összehasonlítás

A téglalap esetén 7.9%-kal jobbnak bizonyult a CMB, az boustrophedon-úttal való bejárás alsó becslésénél. Azért adott ilyen jó eredményt az algoritmust, mert a téglalapnak mind a négy szöge 90°-os, amely nagyobb, mint a szimulációban használt minimálisan bevehető szög.

A paralelogramma és a konkáv sokszög esetén viszont rosszabbnak bizonyult. Az elsőnél 21.7%-kal volt rosszabb a CMB, míg az utóbbinál 3.3%-kal. A paralelogramma két olyan szöget is tartalmaz, amely nem vehető be a kisebb utat eredményező módon, a konkáv sokszögnek viszont csak egy ilyen van, ezért ez jobban megközelítette a párhuzamos egyenesekkel bejárható úthosszt.

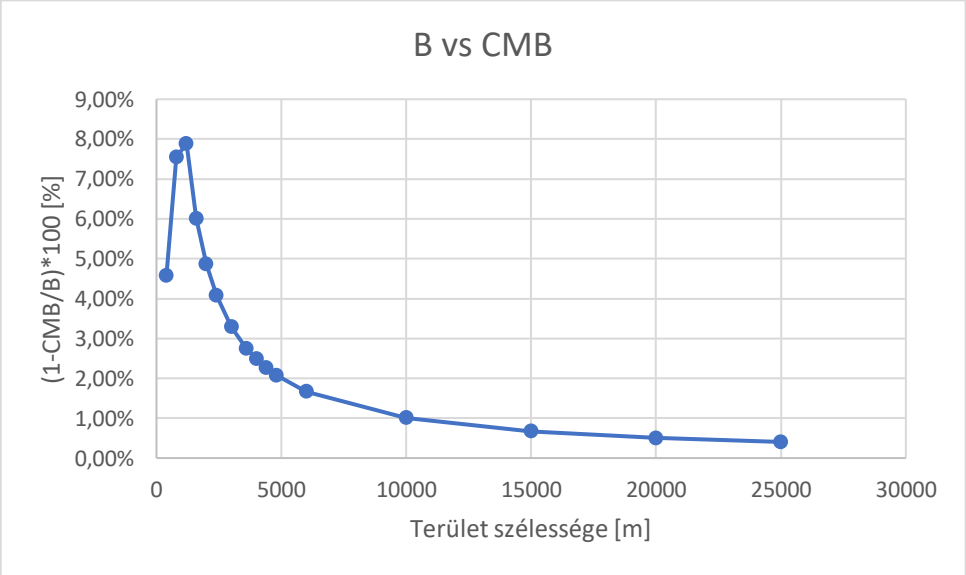
Fontos kiemelni, hogy a párhuzamos egyenesekkel való bejárás esetén egy alsó becsléshez hasonlítunk. Ezt a Boustrophedon módszer elméletben elérheti, de jellemzően ennél nagyobb, így a CMB hatékonysága jelentősen javulhat egy valós esetben.

6.5 CMB hatékonysága a terület növelésével

Érdemes lehet megvizsgálni, hogy a terület növelésével hogyan változik a CMB algoritmus hatékonysága.

A terület jelen esetben egy téglalap. Az magassága mindig 1000 méter, a szélessége pedig 400 méter és 25 000 méter között változik. Ehhez a szimulációs eredményeket a 24. ábra mutatja.

A vízszintes tengely a terület szélességét, a függőleges tengely pedig azt mutatja meg, hogy a CMB hány százalékkal teljesített jobban a Boustrophedon-hoz képest.



24. ábra Terület növekedésével a hatékonyság változása

Megfigyelhető, hogy a területet ahogy növeljük, úgy csökken arányaiban a CMB hatékonysága a Boustrophedon-hoz képest. A maximálisan elért eredmény a 7.78 %, amelyet a szimulációban egy 1000 x 1200 méteres területen való bejárás eredményezett.

6.6 Konklúzió

Összességében elmondható, hogy azon sokszögek esetén, ahol minden szög a minimálisan bevehető szögnél nagyobb, az algoritmus hatékony, ellenkező esetben viszont kevésbé.

Mivel a terület ismert, és nem egy felderítési feladatról van szó, ezért a bejárás előtt egyértelműen eldönthető, hogy érdemes-e alkalmazni a CMB algoritmust.

7 Kooperatív területfedés

Annak érdekében, hogy azon alakzatokra is egy jó alternatív megoldást tudjon adni az algoritmus, amelyben van olyan szög, amely nem éri el a minimálisan bevehető szöget, egy másik járművet is használhatunk.

Egy fajta cella-dekompozícióval megpróbálja az algoritmus szétdarabolni az alakzatot olyan módon, hogy legyen egy „nagyobb” terület, amelyen a Dubins jármű minden probléma nélkül végig tud menni, és a fennmaradó részeket, pedig egy kisebb, de szabadabb mozgású (omnidirekcionális) robottal fedi le. Ennek következtében a CMB hatékonyabb bejárást tud biztosítani a problémás sokszögek esetén is.

A fejezet egy általános és kooperációs pályatervezési összefoglalást is tartalmaz annak érdekében, hogy egy nagyobb képet adjon a kooperatív területfedési problémáról.

7.1 Omnidirekcionális robot bemutatása

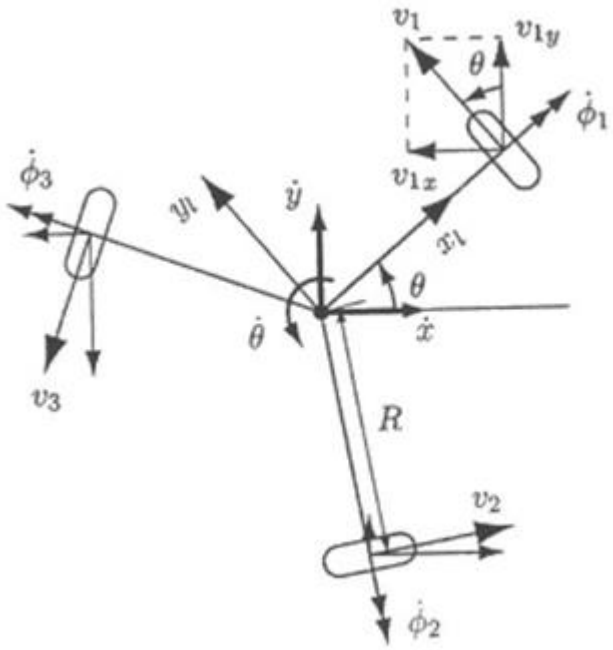
Az omnidirekcionális járműveknek az a nagy előnye a differenciális vagy a Dubins-járművekkel szemben, hogy pillanatszerűen tudnak irányt változtatni, tehát nincs minimális fordulási sugaruk, és nem kell egyhelyben forogniuk ahhoz hogy irányt tudjanak változtatni.

Erre a földön mozgó omnidirekcionális robotoknak úgynevezett omnikerekek adnak lehetőséget (25. ábra).

25. ábra Omnikerék [15]

Valamennyi kerekén görgők helyezkednek el (a 25. ábra által bemutatott kerék esetén, 6 darab görgő, 3-3 mind a két oldalon), amelyek segítségével tetszőleges irányba tud elmozogni a robot.

Az robot állapotát a $(\mathrm{x}, \mathrm{y}, \theta)$ számhármas jellemzi. Az x, y a globális, az x_l és y_l pedig a lokális (robothoz rendelt) koordinátarendszer tengelyei. A lokális koordinátarendszer középpontja a három kerék forgástengelyének metszéspontja (a kerék miatt most feltételezzük, hogy a robot egy földön mozgó jármű). A kerekeknek a távolsága a metszéspontól R, a sugaruk r. Az egyes keréktengelyek által bezárt szög 120° ezért, ha az egyik szögét $\alpha_1 = 0°$-nak definiáljuk, onnan a másik két tengely óramutató járásával ellentétesen számozva (pozitív irányba) $\alpha_2 = 120°$ és $\alpha_1 = 240°$. A kerekekhez rendelhetünk sebességet (v_1, v_2, v_3), valamint szögsebességet is (szög változása: $\dot{\Phi}_1, \dot{\Phi}_2, \dot{\Phi}_3$). A jelölések szemléltetése látható a 26. ábra.



26. ábra Omnidirekcionális robot modellje [15]

A differenciális és a Dubins-járművekhez (3.1) hasonlóan itt is felírhatjuk a mozgásegyenleteket [15]:

$$\begin{pmatrix} \dot{x} \\ \dot{y} \\ \dot{\theta} \end{pmatrix} = \begin{pmatrix} -\sin\theta & \cos\theta & R \\ -\sin(\theta + 120°) & \cos(\theta + 120°) & R \\ -\sin(\theta + 240°) & \cos(\theta + 240°) & R \end{pmatrix}^{-1} \begin{pmatrix} v_1 \\ v_2 \\ v_3 \end{pmatrix}$$

Ha a levegőben mozgásról beszélünk, akkor például egy drónt tekinthetünk omnidirekcionális robotnak.

7.2 Elméleti háttér

Egy az általános mozgás tervezéssel kapcsolatos elméleti háttér szükséges ahhoz, hogy a kooperatív probléma fogalmai megfelelően érthetőek legyenek. Az általános áttekintés után megvizsgálásra kerül, miként használható a játékelmélet a járművek mozgástervezésében.

A játékelméleti fogalmak abban fognak segíteni, hogy meghatározzuk azt, hogy mekkora területet járjanak be az egyes járművek.

7.2.1 Játékelméleti áttekintés, mozgástervezéshez való használhatósága

A játékelméleti kutatások az 1930-as évektől folynak, azóta számos szakirodalom született ebben a témakörben[16][17][18]. Az alap ötlet Morgenstein és Neumann Jánosnak köszönhető.

A játékelméleti megfontolások alapvető célja, hogy egy adott szituációban az optimális megoldást megtaláljuk, a költségeket, erőforrásokat minimalizáljuk. Többszemélyes döntéshozatallal foglalkozik, ennek legegyszerűbb formája a kétszemélyes játékok, de tetszőlegesen sok döntéshozó (vagy játékos) lehet. A döntéshozók számán kívül az egyes játékokat csoportosíthatjuk aszerint, hogy statikus vagy dinamikus problémáról van szó, és aszerint is, hogy kooperatív vagy pedig nem kooperatív a játék.

Dinamikus játékról beszélünk, ha a döntéshozók döntési sorrendje nem semleges a kimenetre vonatkozóan, ezzel ellentétben pedig, ha tetszés szerinti sorrendben dönthetnek a játékosok azonos kimenetel mellett, akkor a játék statikus. A kooperatív és nem kooperatív játékok elnevezése beszédes: a kooperatív játéknál a játékosok költségének összege kerül minimalizálásra, míg a nem kooperatívnál minden döntéshozó a maga költségét szeretné a lehető legkevesebbnek tudni.

A robotikában a pályatervezési problémákat nem mindig egyszerű formalizálni. A játékelmélet egy olyan eszközt ad a kezünkbe, amely segítségével a probléma matematikailag formalizálható, valamint az optimális megoldás megtalálható, továbbá az egyes esetek közötti különbségek számszerűsíthetőek. Éppen ezért a robotikában gyakran használatos játékelméleti megfontolások alapján megoldani az egyes problémákat.

7.2.2 Alapvető játékelméleti fogalmak, jelölések

A döntéshozók vagy játékosok száma legyen N, amelyre igaz, hogy $N \in \mathbb{N}$. Ez a jelen esetben a robotok számát jelöli, tehát $\{A^1, A^2, \dots, A^N\}$ a játékosok halmaza.

Jelölje K a játék szintjeinek számát, ahol $K \in \mathbb{N}$. A játék állapottere legyen X, és a játék állapota k. szinten $x_k \in X$. Itt a szintek a Δt időpontokként előforduló robotok helyzetét jelöli.

Az U_k^i a k. szinten az i. játékos akcióhalmaza. Ezen halmaz egy eleme az u_k^i, amely k. szinten az i. játékos akciója. Egy robot mozgástervezése feladatnál jelentheti azt például, hogy az adott szinten mozog, vagy egyhelyben marad.

Az f_k minden $k \in K$-re az állapotátmeneti függvény. Ekkor a következő állapotot a $x_{k+1} = f_k(x_k, u_k^1, \dots, u_k^N)$ kifejezés megadja.

A Γ_k^i az a halmaz, amely k. szinten az i. játékos számára lehetséges stratégiák halmazát jelöli. A $\gamma_i = \{\gamma_1^i, \gamma_2^i, \dots, \gamma_K^i\}$ az i. döntéshozó stratégiája. A γ minden játékos stratégiáját tartalmazza, ez a játék stratégiája, a $\Gamma = \Gamma^1 \times \Gamma^2 \times \dots \times \Gamma^N$ a játék stratégiájának terét jelöli ($\gamma \in \Gamma$).

A L^i valós értékű függvény adott minden játékosra, és az adott játékos költségfüggvényét jelenti.

7.2.3 Játékelméleti megfontolások több robot esetén

Ahogy eddig is, tegyük fel, hogy minden A^i robot egy merev test, amely képes az R^2 vagy R^3 munkatérben mozogni.

Minden A^i robot célja az, hogy megtalálja az u^i beavatkozó függvényt, amely segítségével elérheti az x_{goal}^i konfigurációt az x_{init}^i kezdő konfigurációból, mindezt az $L^i(x_{init}, x_{goal}, u^1, \dots, u^N)$ költségfüggvény minimalizálása mellett. Az A^i robot stratégiája legyen γ^i. Másképpen $u^i(t) = \gamma^i(x, t)$, tehát megadja, hogy egy adott állapotban és időpontban milyen beavatkozást szükséges választani. A $\gamma = \{\gamma^1, \gamma^2, \dots, \gamma^N\}$ legyen a stratégia, Γ pedig a megengedett stratégiák halmaza.

Ha x_{init} és γ adott, akkor $x(t)$ meghatározható, valamint ha x_{init} és x_{goal} adott, akkor a $L^i(\gamma)$ jelölés használható a $L^i(x_{init}, x_{goal}, u^1, \dots, u^N)$ helyett, és a $L^i(\gamma)$-t tekintjük az A^i robot költségfüggvényének γ stratégia esetén.

Általánosságban elmondható, hogy több olyan $\gamma \in \Gamma$ lesz, melynek a költsége megegyezik. Ezért bevezetésre kerül egy $\sim_L$ jelölés. Akkor és csak akkor mondjuk, hogy $\gamma \sim_L \gamma'$(γ stratégia ekvivalens γ' stratégiával), ha igaz, hogy $L^i(\gamma) = L^i(\gamma')$ $\forall i$-re.

Az ekvivalencia reláció a Γ osztályt olyan részosztályokra bontja, amely stratégiáknak a költsége megegyezik. Ezek az osztályok alkotják a tört stratégiák terét, amely a $\Gamma/\sim$ jelölést kapta. A $\Gamma/\sim$ egy eleme egy tört stratégia, jelölése $[\gamma]_L$. Ez azt a tört stratégiát jelöli, amelynek a része a γ stratégia.

Több robot esetén a cél az, hogy az összes robot együttes költségét minimalizáljuk, kooperatív játékot feltételezünk. A célt egy kicsit másképpen megfogalmazva, keressük azokat a stratégiákat, amelyek a minimális költséget eredményeznek. Ezek lesznek a minimális stratégiák.

A $\Gamma/\sim$ térben egy rendezést definiálunk ($\preccurlyeq$), amely a stratégiák között teremt kapcsolatot. Azt mondhatjuk, hogy $\gamma \preccurlyeq \gamma'$, ha $L^i(\gamma) \leq L^i(\gamma')$ $\forall i$-re. Ha ezen kívül az is igaz, hogy $L^j(\gamma) < L^j(\gamma')$ valamely j-re, akkor azt mondjuk, hogy a $[\gamma]_L$ stratégia jobb, mint $[\gamma']_L$. Két stratégia nem összehasonlítható, ha létezik olyan i és j, hogy $L^i(\gamma) < L^i(\gamma')$ és $L^j(\gamma) > L^j(\gamma')$.

Azt mondjuk, hogy $[\gamma^*]_L$ minimális stratégia, ha minden $[\gamma]_L \neq [\gamma^*]_L$-re, ahol a két stratégia összehasonlítható, igaz, hogy $[\gamma^*]_L \preccurlyeq [\gamma]_L$.

A kooperatív játékelmélet egy népszerű optimális koncepciót használ, amely a Pareto optimum. Egy γ^* stratégia Pareto optimum, ha bármely más $\gamma \in \Gamma$ stratégia esetén egyik robot sem tud a költségén csökkenteni úgy, hogy más robotnak a költsége ne növekedjen. A minimális stratégiák Pareto optimumok is egyben.

Nem kooperatív játékok esetén a Nash egyensúly egy népszerű optimum, az itt tárgyalt feladat során nem alkalmazandó, de nagy jelentőségű, ezért mindenképp megemlítendő [19][20].

Megvizsgálható a skalár értékű optimalizálás és a minimális stratégiák kapcsolata. A skalár értékű optimalizálás célja egy olyan leképezés meghatározása, amely a veszteségeket skaláris értékre vetíti, miközben garantálja, hogy a skalárveszteség optimalizálása minimális stratégiát eredményez.

Legyen $\beta = \{\beta_1, \beta_2, \dots \beta_N\}$ pozitív valós számokból álló halmaz, amelyre igaz. hogy

(7.1)

$$\sum_{i=1}^{N} \beta_i = 1$$

Ennek segítségével a költségfüggvény skalárrá alakítása:

$$H(\gamma, \beta) = \sum_{i=1}^{N} \beta_i L^i(\gamma)$$

A skalár költségfüggvényből látható, hogy az egyes robot között priorizálni lehet a β értékének módosításával. Ha a $\beta_i = \frac{1}{N}$ értéket választjuk $\forall i \in \{1, 2 \dots, N\}$ futóváltozóra, akkor minden robotot azonos súllyal számolunk.

Tételként kimondható, hogy ha β egy fix, rögzített érték, és γ^* egy olyan stratégia, amely minimalizálja a $H(\gamma, \beta)$ függvényt, akkor a $[\gamma^*]_L$ tört stratégia minimális.

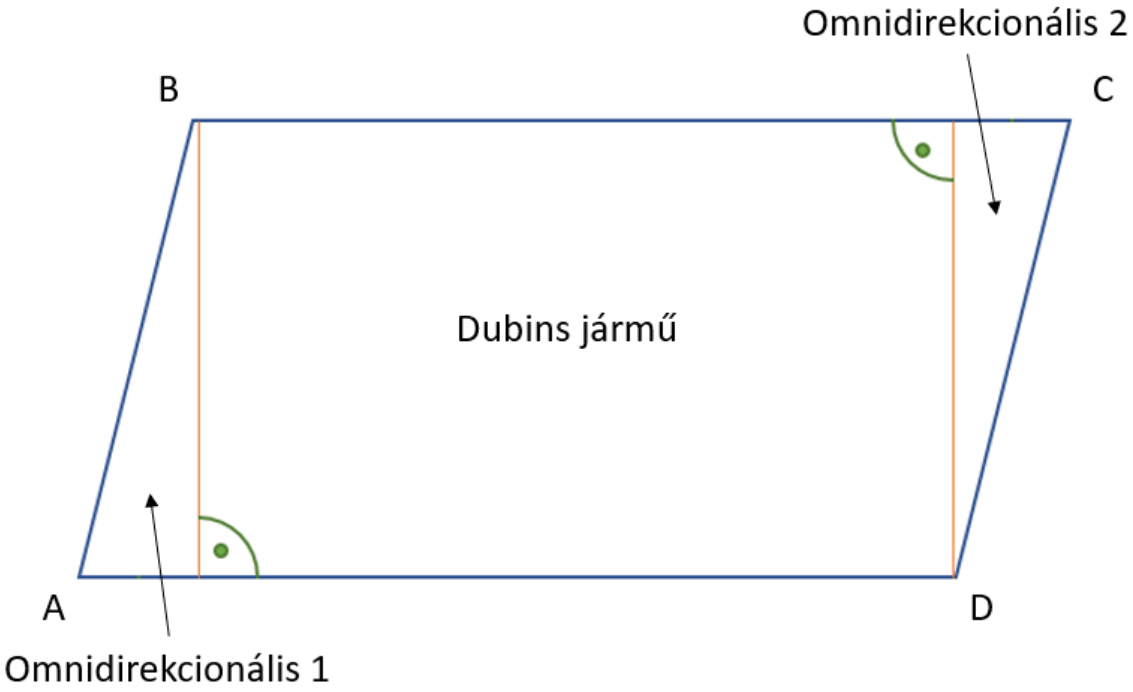
7.3 Kooperatív területfedés megvalósítása

7.3.1 Cella-dekompozíció a kooperatív területlefedésben

Olyan alakzat darabolásra van szükség, amely a kis szögeket eliminálja a Dubins-jármű által bejárni szükséges alakzatból, és helyette egy a minimálisan bevehető szögnél nagyobb szög jön létre.

Ha 90°-ot meghaladja a minimálisan bevehető szög, akkor az algoritmus értelmét veszti, mivel nagyon speciális eset kellene hozzá, hogy minden szög 90°-nál nagyobb legyen (például szabályos hatszög). Ezért feltételezzük, hogy ha az algoritmus akkor használható, ha ez a minimálisan bevehető szög nem haladja meg a 90°-ot.

Ezen megfontolások alapján a cella dekompozíció olyan módon történik, hogy a „kis” szögeknél (27. ábra A és C csúcsoknál) a csúcshoz közelebbi szomszédos csúcsból (kivéve, ha a szomszédos csúcs is „kis” szögű, jelen példában rendre B és D), merőlegest állítunk a másik szomszédos és a „kis” csúcsot összekötő szakaszra (rendre AD és BC oldalakra).



27. ábra Cella dekompozíció szemléltetése

Ezzel a Dubins-járműnek a területe számára könnyen bejárható, míg a kisebb szögű problémás részeket pedig az omnidirekcionális jármű járja be. Az omnidirekcionális bejárása során a területe leghosszabb oldalával párhuzamosan halad végig majd, ha több területet is be kell járnia, akkor átmegy a másik terület leghosszabb oldalához, és ott folytatja a bejárást.

Az egyes robotok mozgásának időbeliségét is megnézve figyelembe kell venni az ütközés lehetőségét, amelyet játékelméleti eszközökkel lehet kiküszöbölni.

7.3.2 Szimulációs eredmények

A téglalapnál a cella-dekompozíciónak nincs jelentősége, ez csak abban az esetben alkalmazandó, ha van a minimálisan bevehető szögnél kisebb a területen.

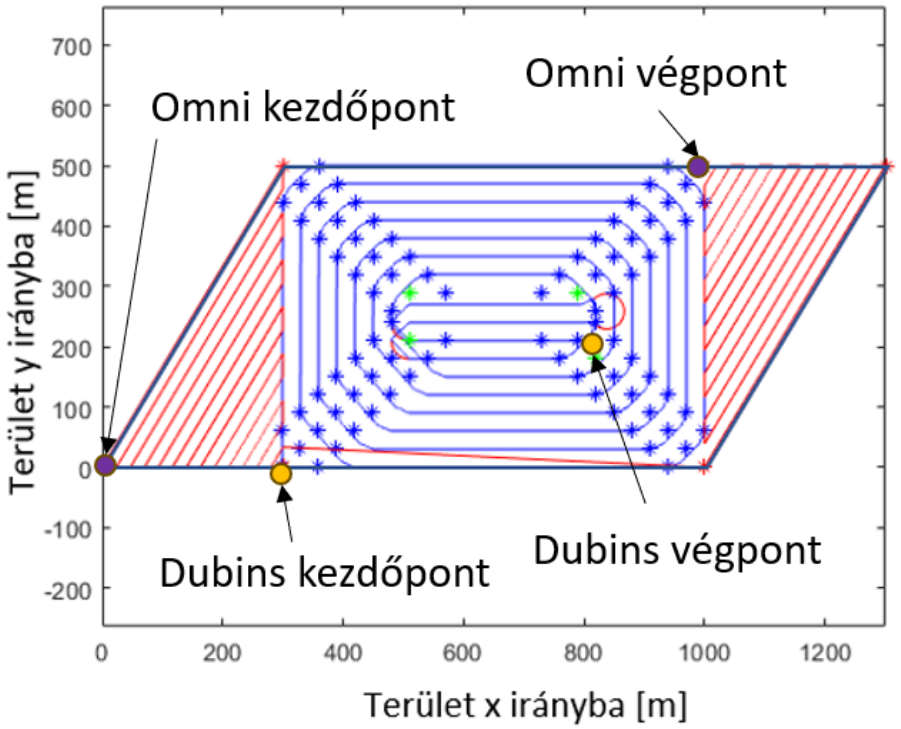
A paralelogramma és a konkáv esetben azonban megvizsgálandó, hogy növelte-e, ha igen mennyivel az algoritmus hatékonyságát.

A szimuláció során az omnidirekcionális jármű lefedésének képességét (w_{omni}) a Dubins- jármű kétharmadának vettem.

7.3.2.1 Paralelogramma

A paralelogramma esetén az algoritmus levág két háromszöget a területből, így a Dubins-járműnek egy téglalapot kell bejárnia, míg az omnidirekcionális robotnak a fennmaradó részt.

A területlefedést a 28. ábra mutatja be, kék színnel a CMB, piros színnel az omnidirekcionális útvonala látható.

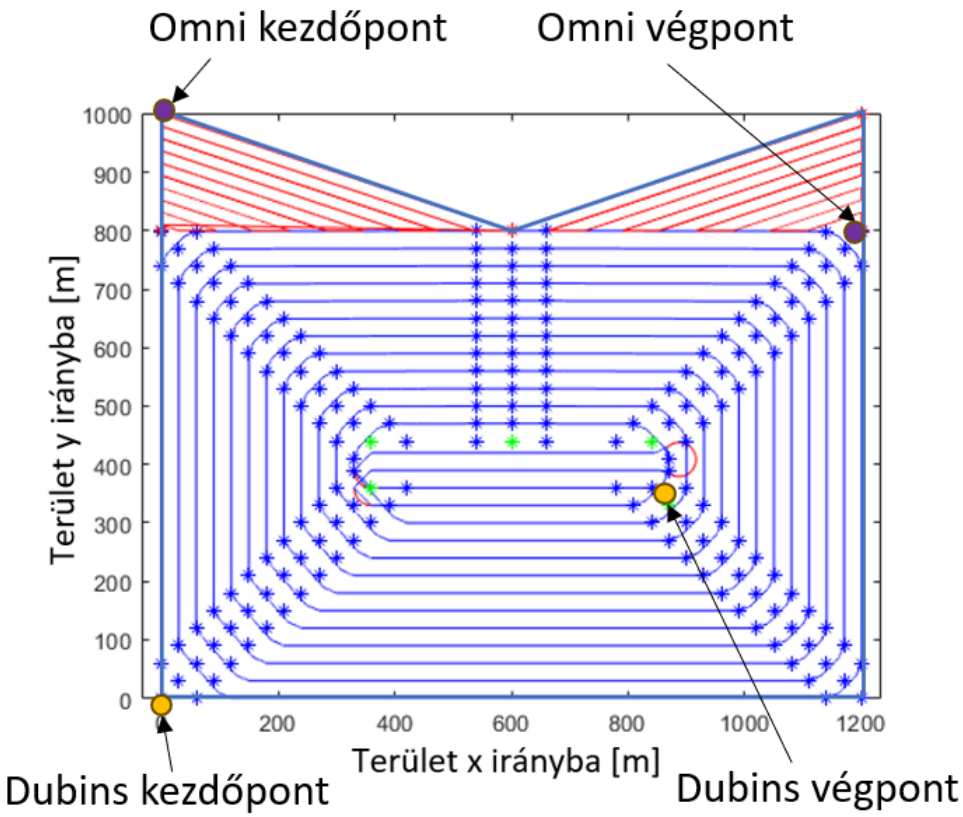


28. ábra Paralelogramma kooperatív területfedése

7.3.2.2 Konkáv sokszög

Ebben az esetben két szög nem felelt meg a kritériumoknak. Alapvetően a CMB algoritmusban csak egy szöget nem tudott bevenni a jármű, viszont a cella-dekompozíció a pontgenerálások előtt történik meg, és csak az eredeti sokszög szögeit tekinti azt nem, hogy a generált pontok hogyan változtatják meg a szögeket.

Az eredményeket a 29. ábra mutatja.



29. ábra Konkáv sokszög kooperatív területfedése

7.3.2.3 Kooperáció összehasonlítás

A kooperatív területlefedés mérőszáma is a megtett út, bár ez igen nagy kihívások elé állítja az algoritmust. Az omnidirekcionális robot lefedési képessége kétharmada a Dubins-járműének, de a dekompozíció nem csökkenti a területet, csupán részekre osztja. Ezért az egész levágott (piros) területnek a bejárása nem lehet több, mint amennyi a Dubins-járműnek kellene, hogy bejárja a piros területet (egyenesekkel és forgásokkal együtt), amely kétharmad akkora sugárral nem egyszerű. Ez tekinthető egy elérendő célnak.

Az eredményeket a 3. táblázat tartalmazza, melyek megmutatják, hogy ha a megtett utat tekintjük mérőszámnak, akkor a cella-dekompozíció nem hozott nagy áttörést.

A paralelogramma esetén 17%-kal rosszabb, mint a boustrophedon-utas bejárás alsó becslése, a CMB algoritmusnál viszont 3.7%-kal jobb eredményt adott. Ha a konkáv alakzatot nézzük, az nagyobb bizakodásra ad okot; a boustrophedon-utas bejárásnál 7.7%-kal volt jobb, míg a CMB algoritmusnál 10.6%-kal.

	Paralelogramma	Konkáv sokszög
CMB úthossza [m]	22 355	45 153
Boustrophedon alsó becslés úthossza [m]	18 373	43 730
Kooperációs Dubins úthossza [m]	11 998	32 358
Kooperációs omnidirekcionális úthossza [m]	9 536	8 008
Kooperációs hossz összesen [m]	21 534	40 366
Koop. Dubins/CMB	0.537	0.717
Koop. összes/CMB	0.963	0.894
Koop. Dubins/B	0.653	0.739
Koop. összes/B	1.172	0.923
Új terület/régi terület	0.7	0.944

3. táblázat Kooperációs módszer összehasonlítás

Azonban a táblázatban fel van tűntetve, hogy a jármű bejárt útja mennyivel csökkent az egyes esetekben, és ez mennyivel kevesebb bejárt területet jelent (új terület/régi terület). Mindkét alakzatnál igaz, hogy a CMB algoritmushoz képest arányaiban kevesebb utat tett meg a Dubins-jármű, mint amennyi terület maradt neki. A paralelogramma esetén a kezdeti terület 70%-át kellett bejárnia, és 53.7%-át tette meg az eredeti úthossznak. A konkáv alakzat esetén a terület 94.4%-át kellett bejárnia és ezt 71.7% úttal tette meg. A konkáv alakzat esetén a boustrophedon bejáráshoz képest is tudott arányaiban a Dubins-jármű javítani, ott 73.9%-nyi utat járt be (miközben a terület a 94.4%-ára csökkent).

A számokból megállapítható, hogy a kooperációs területlefedés egyértelműen lehet jó irány, a CMB algoritmushoz képest mindkét alakzatnál tudott javítani, azonban a

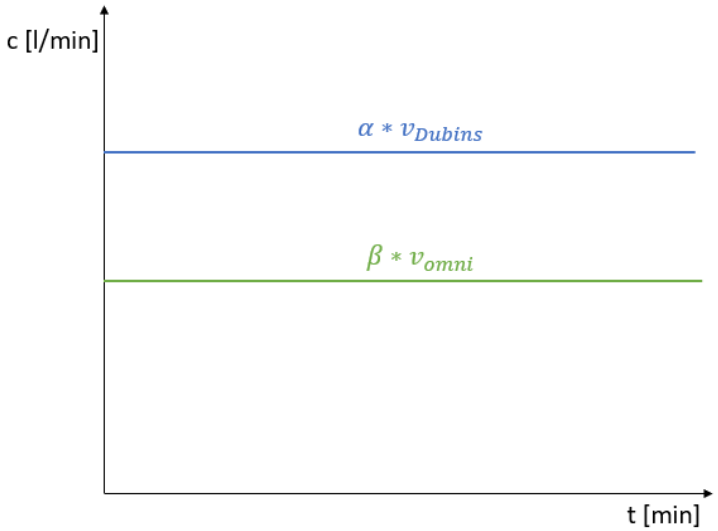
boustrophedon-utas bejárást tekintve még az elvi minimumot nem minden esetben közelíti meg.

7.4 Mérőszámok a kooperációs eredményekhez

Szeretnék definiálni néhány mérőszámot, amely a továbbiakban szükséges ahhoz, hogy az eredményeket értékelni lehessen, nem csupán az általuk megtett út alapján.

A költség mértékegysége legyen a liter, amely az eszközök által az út megtétele során elfogyasztott üzemanyag (feltételezzük, hogy mind a ketten azonos árú üzemanyaggal működtethetők, és így a költségüket ugyanakkora értékkel kell megszorozni, hogy bármiféle pénznemre átváltható legyen a végeredmény).

Legyen a Dubins-jármű fogyasztása α, az omnidirekcionálisé pedig β liter/kilométer (l/km) ($\alpha, \beta \in \mathbb{R}^+$). Ha a sebességük v_{Dubins} és v_{omni} (km/h), akkor $c_{Dubins} = \alpha * v_{Dubins}$ és $c_{omni} = \beta * v_{omni}$ mértékegysége liter/óra (l/h) vagy liter/perc (l/min), amely értékek csak a sebességtől függenek szóval, ha állandó mozgási sebességet feltételezünk, akkor az idő elteltével nem változnak. Tehát az $c(t)$ függvényük egy konstans lesz, amelyet a 30. ábra bemutat.



30. ábra Az játékosok költsége egységnyi idő alatt az idő függvényében

Az α és β értékei természetesen nem feltétlen az ábrán látható kapcsolatban állnak, ezeket a szemléltetés kedvéért választottam így meg.

Ezen függvényeket, ha idő szerint integráljuk, megkapjuk az adott játékos költségét. A Dubins-jármű mozgási ideje legyen T_{Dubins}, az omnidirekcionálisé pedig T_{omni}. Ekkor:

$$L_{Dubins} = \int_0^{T_{Dubins}} \alpha * v_{Dubins}\, dt$$

$$L_{omni} = \int_0^{T_{omni}} \beta * v_{omni}\, dt$$

Ezek alapján egy elsőfokú lineáris függvény lesz a játékosok költségfüggvénye (sebességváltozáskor a meredekség változik).

Most tekintsük a 7.3.2. alfejezetben található eredményeket. Az itt felvázoltak alapján meghatározható, hogy milyen α és β esetén lesz a kooperációs módszer előnyös.

A szimuláció során használt paraméterekkel a költség (a paralelogrammát tekintve):

	Dubins-jármű	**Omnidirekcionális jármű**	**Összesen**
Költség CMB	22.35α	-	22.35α
Költség B	18.375α	-	18.375α
Költség koop	12α	9.54β	$12\alpha + 9.54\beta$

4. táblázat Kooperációs mozgás költsége

Ebben az esetben akkor éri meg a kooperációs módszert alkalmazni a párhuzamos egyenesek helyett, ha $12\alpha + 9.54\beta < 18.375\alpha$, tehát ha $\beta < 0.668\alpha$.

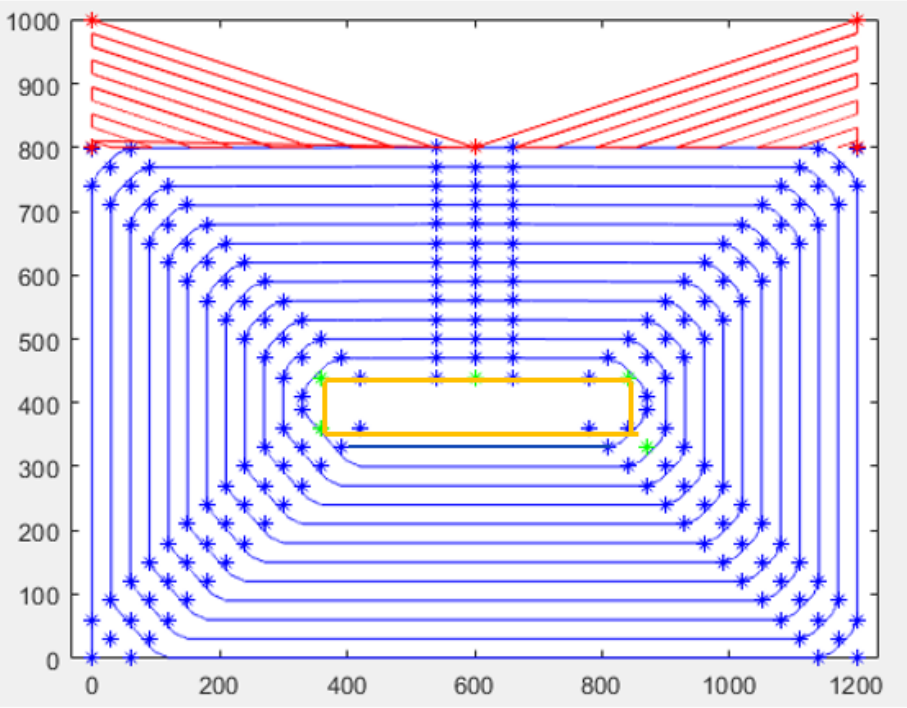
7.5 Stratégiák vizsgálata a kooperációs területfedésben

A kooperációs feladat során az lesz a feladat, hogy megtaláljuk a minimális γ stratégiát, tehát azt a bejárást, ahol a $L(\gamma)$ költségfüggvény minimális. A robotok fix pálya mentén mozognak, tehát a különböző stratégiákat az fogja meghatározni, hogy hol van az a x_{init} és x_{goal} pontok, ahol az egyes robotok a mozgásukat elkezdik és befejezik. Az $u^1, \dots, u^N$ akciókat az algoritmus előre meghatározza, ennek a meghatározására külön nincs szükség. A játékosok száma továbbra is $N = 2$.

Két szempontból hasonlítok össze két különböző stratégiát.

A két szempont abban különbözik, hogy mit szeretnénk minimalizálni. Az út és az idő optimalizálás szemszögéből fogom a stratégiákat vizsgálni.

Az első (Γ_1) stratégia az eddig bemutatott kooperációs területfedés (7.3 alfejezet). A második (Γ_2) stratégia pedig abban különbözik a Γ_1 stratégiától, hogy az omnidirekcionális robot fedi le azt a területet is, amit a Dubins-jármű a CMB algoritmusban párhuzamos egyenesekkel fedett le (31. ábra narancssárga terület).



31. ábra Γ_2 terület elosztása

A stratégiákat a két szempont szerint egy paralelogrammán és a már bemutatott konkáv alakzaton fogom vizsgálni.

7.5.1 Út, mint költség

Ebben a alfejezetben azt vizsgálom, hogy hogyan alakul az út a két stratégia esetén. Az eredményeket az 5. táblázat mutatja.

A Γ_2 stratégia értékei úgy számolhatóak, hogy a Dubins-jármű útjából levonásra került, az az út, amelyet a 31. ábra szerint bejelölt narancssárga részen tett meg. Az omnidirekcionális robot útjához pedig hozzáadódott az az út, ami ahhoz kell, hogy eljusson ehhez a területhez (az eredetileg neki szánt terület befejezése után), majd azt a hosszabbik oldal mentén párhuzamos egyenesekkel bejárja.

(Dubins + omnidirekcionális)	**Paralelogramma**	**Konkáv alakzat**
$\boldsymbol{\Gamma_1}$ **stratégia költsége [m]**	11 998 + 9 536 = 21 534	32 358 + 8 008 = 40 366
$\boldsymbol{\Gamma_2}$ **stratégia költsége [m]**	10 752 +11 230 = 21 982	30 528 + 10 918 = 41 446

5. táblázat Út, mint költség

Látható, hogy mind a két esetben a Γ_1 stratégia lett a jobb. Többel nőtt az omnidirekcionális jármű úthossza, mint a Dubins-é csökkent.

7.5.2 Idő, mint költség

Ebben az esetben azt vizsgálom, hogy ha a lefedés költségének az időt tekintjük, miként alakul a költség. Ez egy olyan játék, ahol mind a két játékos maximalizálni akarja a lefedett területét, ezzel az időt minimalizálni.

A számolásokban a $v_{Dubins} = 25\ \frac{m}{s}$, $v_{omni} = 15\ \frac{m}{s}$ és a sebességet állandónak tekintem. A max() függvény a két elem maximumát választja ki. Ez azért használandó, mert a két bejárás idejét tekintve a több idő lesz az egész terület bejárásának ideje.

max(Dubins; omnidirekcionális)	**Paralelogramma**	**Konkáv alakzat**
$\boldsymbol{\Gamma_1}$ **stratégia**	max(480; 636) = 636	max(1 294; 534) = 1 294
$\boldsymbol{\Gamma_2}$ **stratégia**	max(430; 748) = 748	max(1 221; 727) = 1 221

6. táblázat Idő, mint költség

A paralelogramma esetén a Γ_1 stratégia lett az eredményesebb, viszont a konkáv alakzat esetén pedig a Γ_2. A konkáv alakzatnál látható, hogy a Dubins-jármű és az omnidirekcionális robot területfedéshez szükséges ideje között van 497 másodperc. Ebben az esetben az omnidirekcionális járművet érdemes lehet elindítani a Dubins-jármű útján, és megkeresni azt a pontot, amikor a bejáráshoz szükséges idő közel azonos lesz (de erre a vizsgálatra a dolgozat nem terjed ki).

8 Összefoglalás

A dolgozat keretein belül a Dubins-járművek területfedésének problémájával foglalkoztam. Azt vizsgáltam, hogy ha a robot nem párhuzamos egyenesekkel fedi le a területet, hanem helyette először csigaszerű módon kezdi meg a lefedést, akkor miként javíthatóak a bejárás mutatói. Az algoritmus a CMB elnevezést kapta.

A szimulációs eredmények szerint az algoritmus azokban az esetekben, amikor egy a Dubins-jármű paramétereitől függő minimálisan bevehető szögnél a bejárandó terület minden szöge nagyobb, hatékonyan működik. Kevesebb út megtételével járja be a területet, mint a párhuzamos egyenesek módszerének elvi minimuma. Azonban, ha van olyan szög, amely nem éri el a küszöbértéket, az algoritmus hatékonysága romlik.

Erre kínál megoldást a terület szétdarabolása, mely során egy omnidirekcionális robotot is alkalmazunk a lefedés során annak érdekében, hogy a problémás szögeket ő fedje be. Ezzel a módszerrel a CMB algoritmuson sikerült javítani, valamint jobban megközelíteni a párhuzamos egyenesek bejárásának útvonalhosszát.

Ha nem csak az útvonalat tekintjük költségnek, hanem egy általánosabb fogyasztást definiáló konstanssal számolunk, akkor a kooperációs megoldás ígéretesebb eredményt ad. Arányaiban javult a Dubins-jármű bejárása, bár kevesebb területet fedett le, cserébe lényegesebben kevesebb utat kellett ehhez megtennie.

Végül vizsgáltam, hogy milyen stratégiát érdemes használni, ha az utat vagy a bejáráshoz szükséges időt tekintjük költségnek.

Továbbfejlesztési lehetőségnek gondolom a kooperáció során a több robot használatának vizsgálatát. Akkor lehetséges utat spórolni, ha a különböző területeket különböző járművek fedik le, ekkor az egyik területről a másikra való áthaladást nem kell megtenniük. Ezen kívül vizsgálnám az algoritmus hatékonyságát olyan esetben is, ha van tiltott zóna a területen, tehát egyéb korlátozások is jelen vannak. Továbbá a kooperációs területfedésnél több stratégiát is érdemes lehet megvizsgálni, hogy az egyes mérőszámokat csökkenteni tudjuk.

Irodalomjegyzék

[1] Chengmin Zhou, Bingding Huang, Pasi Fränti: *A review of motion planning algorithms for intelligent robots,* Journal of Intelligent Manufacturing (2022) 33:387–424

[2] Yaru Kang, Dianxi Shi: *A Research on Area Coverage Algorithm for Robotics, published in* 2018 IEEE International Conference of Intelligent Robotic and Control Engineering (IRCE), DOI: 10.1109/IRCE.2018.8492964, 2018

[3] Xin Yu: *Optimization Approaches for a Dubins Vehicle in Coverage Planning Problem and Traveling Salesman Problems,* Auburn University, 2015

[4] J. Lewis, W. Edwards, K. Benson, I. Rekleitis, and J. O'Kane, "*Semiboustrophedon coverage with a dubins vehicle*," in Proc. IROS, 2017

[5] M. Waanders, Coverage path planning for mobile cleaning robots, University of Twente, 2011

[6] Abdul Majeed, Seong Oun Hwang: *A Multi-Objective Coverage Path Planning* Algorithm for UAVs to Cover Spatially Distributed Regions in Urban *Environments,* Department of Computer Engineering, Gachon University, 2021

[7] Amna Khan , Iram Noreen , Zulfiqar Habib: *On Complete Coverage Path Planning Algorithms for Non-holonomic Mobile Robots: Survey and Challenges,* Department of Computer Science, COMSATS Institute of Information Technology, Lahore

[8] Wesley H. Huang: *Optimal Line-sweep-based Decompositions for Coverage Algorithms,* Proceedings of the 2001 IEEE International Conference on Robotics & Automation Seoul, Korea 2001

[9] Theresa Marie Driscoll: *Complete coverage path planning in an agricultural environment*, Graduate Theses and Dissertation, Iowa State University Capstones, 2011

[10] J. Jin and L. Tang, "Optimal Coverage Path Planning for Arable Farming on 2D Surfaces," Transactions of the ASABE, vol. 53, no. 1, pp. 283-295, 2010.

[11] H. Choset, "*Coverage of known spaces: The boustrophedon cellular decomposition*" Autonomous Robots, vol. 9, no. 3, pp. 247–253, 2000.

[12] Matthew Coombes, Wen-Hua Chen, Cunjia Liu: *Boustrophedon Coverage Path Planning for UAV Aerial Surveys in Wind,* Department of Automotive and Aeronautical Engineering Loughborough University, Loughborough, 2018

[13] Steven M. LaVelle: *Planning Algorithms*, Cambridge University Press 2016, http://planning.cs.uiuc.edu/node821.html

[14] W.H Huang: *Optimal line-sweep-based decompositions for coverage algorithms,* IEEE International Conference on Robotics and Automation (Cat. No.01CH37164)

[15] T.A. Baede: Motion control of an omnidirectional mobile robot, Eindhoven, September 18th, 2006, DCT 2006.084

[16] Gincsainé Dr Szédeczky-Kardoss Emese: *Multiágensű robotrendszererk mozgástervezése játékelméleti eszközökkel,* Budapesti Műszaki és Gazdaságtudományi Egyetem, 2009

[17] Steven Michael LaValle: *A game-theoretic framework for robot motion planning,* PhD thesis, Urbana, Illinois, 1995

[18] Dr Harmati István: *Nem kooperatív véges kétszemélyes nulla összegű játékok*, Budapesti Műszaki és Gazdaságtudományi Egyetem, 2009

[19] Zhou FENG:*On the Nash Equilibrium in Game Theory A Learning Report for Optimization Method Class,* November 14, Huazhong University of Science and Tecnology, 2018

[20] Osborne, Martin J.; Rubinstein, Ariel: *A Course in Game Theory,* 12 Jul 1994, Cambridge