Autonóm robotok gyülekezése lokális tömegközéppont alapú algoritmus használatával

Bolla Kálmán[1], Johanyák Zsolt Csaba[1], Kovács Tamás[1], Fazekas Gábor[2]
[1]Kecskeméti Főiskola, GAMF Kar, Informatika Tanszék
[2]Debreceni Egyetem, Informatikai Kar, Információ Technológia Tanszék

Összefoglalás: A mobil robotok összegyűlési módszereinek fejlesztése a swarm intelligencia intenzíven kutatott területe. Cikkünkben egy olyan új algoritmust mutatunk be, amelynek működése az aktuális robot szemszögéből látható robottársak tömegközéppontjának számításán alapszik. A szimulációs futtatások során az új algoritmus a korábbi ismert megoldásoknál jobb teljesítménymutatókat eredményezett.

Abstract: The development of mobile robot gathering algorithms is an intensively investigated research area. This paper presents a new algorithm that is based on the calculation of the center of gravity of the visible neighboring robots. Several simulation runs proved that the suggested algorithm results better performance than some previously published other solutions.

Kulcsszavak: gyülekezési algoritmus, mobil robotok

Keywords: robot swarm, gathering, autonomous mobile robots

1. Bevezetés

Az elmúlt néhány évtizedben a robotika egyik kedvelt kutatási területévé vált a robot swarm feladatok lehetséges megoldásainak kutatása autonóm elosztott rendszerek esetén. A swarm intelligencia alapfeladatra az irodalomban található jelenlegi megoldások általában olyan elméleti eredmények, amelyek a gyakorlatban nem, vagy erős korlátozásokkal alkalmazhatóak. A legintenzívebben kutatott feladatok a következők: robotok egyetlen pontba gyülekezése (vagy a lehető legkisebb területre) [1-11], területbejáró algoritmusok alkalmazása egy bázispontból indulva [13], és hasznos részecskék (pl.: táplálék) begyűjtése különböző megoldásokkal ([12,14]).

Ebben a cikkben a gyülekezés problémájára koncentrálunk akadálymentes környezetben. A gyülekezés eredeti feladatmegfogalmazása szerint tetszőleges alakzatból kiindulva véges idő alatt egyetlen pontban kell a robotoknak találkozniuk. Kézenfekvő megoldás lehet erre a problémára egy egyszerű konvergencia alapú megközelítés, ami feltételezi azon terület átmérőjének ismeretét, ahol a robotok elhelyezkednek. Emellett szükséges, hogy minden robot belássa a teljes teret. Később ezt az átmérőt csökkentjük minden lépésben a gyülekezés sikeres megvalósulásáig. A fentiekben felvázolt megközelítés alkalmazásakor globális érzékelőknek kell segíteniük a robotok tevékenységét.

A cikkünkben ismertetésre kerülő megoldás a fentiektől eltérően csak minimális robotképességeket igényel, és csak lokális információk alapján dönt. Egy olyan algoritmus, ami a robotokon fut autonóm módon, nagyban függ a robotok képességeitől. Az alábbiakban felsoroljuk a feladat megoldása szempontjából fontos képességeket:

- rendelkezik-e memóriával (visszatérhet-e az előző pozícióba),
- szinkron vagy aszinkron működésű,
- globális navigációs képességgel rendelkezik (közös koordináta rendszerrel) vagy sem,

- limitált látótávolsága vagy az egész környezetét belátja,
- tudnak-e kommunikálni a robotok egymással vagy sem,
- pontszerű vagy kiterjedéssel rendelkező robotreprezentációt használunk.

A legtöbb gyülekezési algoritmust feledékeny (memóriával nem rendelkező) robotokra fejlesztették ki globális navigáció használata nélkül. A feledékenység itt azt jelenti, hogy a robot nem tud visszalépni az előző pozíciójába, mivel csak az adott pillanat érzékelése alapján számolja a célvektorát. Az előző lépést nem tudja eltárolni, így az nem befolyásolja a jövőbeni lépések számításában. Ez az egyik olyan alapvető feltétel, amitől gyülekezés problémára nem egy triviális feladat. Tipikusan egy gyülekezési algoritmusban a következő lépések ismétlődnek, amennyiben az egyes lépések szinkronizálva vannak a robotok között:

- *Look*: látható robotok pozíciójának meghatározása
- *Calculate*: következő pozíció számolása a *Look* alapján
- *Move*: mozgás a kiszámított pozíció felé

Szinkron működésnél a robotok egyszerre hajtják végre a soron következő lépést egy szinkronjel hatására vagy belső szinkron segítségével. Belső szinkron esetén nem kell a robotok között a kommunikációt megvalósítani, csupán meg kell állapítani azokat a maximális időintervallumokat, ami alatt a körbenézés, következő lépés számítása, valamint a leghosszabb lépés végrehajtható. Minden robot a megfelelő lépéshez tartozó időtartamot várakozik, még akkor is, ha nem kell lépnie sehova. Egy aszinkron modellben azonban a robotok különböző időpontokban indíthatják a lépéseiket attól függően, hogy mikor fejezték be előző lépésüket. Ebben az esetben egy *Wait* (várakozás) lépés is beékelődik a *Move* és *Look* lépések közé.

Tömegközéppont (COG – Center Of Gravity) számításon alapuló algoritmust Cohen és Peleg [3] használta először, szinkron működésű robot swarm esetén, amely tetszőleges alakzatból kiindulva sikeresen hajtja végre a gyülekezést. Idealizált megoldásunkban egy igen erős feltételezéssel élnek: a robotok látótávolsága nem korlátos, így minden egyed belátja a teljes környezetet.

Cieliebak és társai [2] COG alapú algoritmus helyett a látható robotokra legkisebb ráhúzható kör középpontja felé történő elmozdulást alkalmazták (SEC - Smallest Enclosing Circle), és ezzel az algoritmussal megoldást adtak pontszerű robotok aszinkron gyülekezésére feledékeny robotokat feltételezve. Ebben az esetben sincs látótávolság korlátozás bevezetve.

A fentiekben említett algoritmusok megoldást kínálnak a gyülekezés feladatára pontszerű és nem korlátos látótávolságú robotok esetén. A valós feltételek között is alkalmazható modellek kialakítása irányában tett lépésként értékelhető az Ando és szerzőtársai [1] által pontszerű robotok esetére kifejlesztett szinkronműködésű algoritmus, ahol a robotok csak limitált látótávolsággal rendelkeznek. A limitált látótávolságot feltételező megoldásokban központi szerepet játszik az ún. láthatósági gráf. A gráf csomópontjai jelentik a robotokat, az élek pedig a robotok közötti kapcsolatot abban az esetben, ha azok látják egymást. Habár Ando és szerzőtársai is SEC algoritmust használtak, mégis minden egyed más középpont felé mozdult el, mivel minden robot a közvetlen környezete alapján számolta a következő lépését. Emellett minden robotmozgás korlátozott volt, mivel a sikeres gyülekezéshez elengedhetetlen, hogy a láthatósági gráf ne szakadjon meg. A lépések mértékének ellenőrzése nélkül kisebb csomópontok alakulnának ki, mivel a gráfban található kapcsolatok sérülhetnek. A lokális SEC algoritmusuk működését szinkron esetre bizonyították be.

Később Flocchini [4] és Souissi [6] társaikkal együtt mutattak be limitált látótávolság mellett egy aszinkron megoldást. Azonban itt feltételezték, hogy a robotok az irányultsággal is tisztában vannak (iránytűvel vannak felszerelve), így ők egyfajta globális navigációs rendszert

használtak a feladat megoldásához.

A szakirodalomban az egyik legfrissebb szinkron és limitált látást alkalmazó megoldás Degener nevéhez [10] fűződik, aki a COG és SEC helyett lokális konvex sokszög alapján határozza meg a következő lépést. Ebben a megoldásban nincs globális navigáció, viszont a kommunikáció engedélyezett, így a robotok meg tudják osztani egymással jövőbeni lépéseiket, jelenlegi pozíciójukat.

A gyakorlati megvalósítás érdekében tett következő lépés a pontszerű robotreprezentáció elhagyása, azaz kiterjedéssel rendelkező robotreprezentáció használata. Ebben az esetben a robotokat zárt alakzatként értelmezzük, ami egy középponttal és R_s sugárral rendelkezik. Azonban ennek a módosításnak komoly következményei vannak: a robotok nem képesek egyetlen pontba gyülekezni, így az eredeti gyülekezési definíciót meg kell változtatnunk. Másik probléma, hogy egy robot blokkolni tudja egy másik robot mozgását, valamint egy robot takarhatja több társát is a megfigyelő szemszögéből (mivel az egyedek nem átlátszóak) hiába helyezkednek el a láthatósági sugáron belül.

Ezen a ponton újra kell definiálni a gyülekezés fogalmát kiterjedéssel rendelkező robotok esetében. Czyzowicz és társai [8] ezt a következőképpen határozták meg:

- a zárt alakzatok kontakt gráfja összefüggő, és
- mindegyik robot látja a többi robotot.

Hozzá kell tennünk, hogy ők legfeljebb 4 robotra adták meg a gyülekezést, viszont több robot esetén a definíció második része már nem teljesíthető.

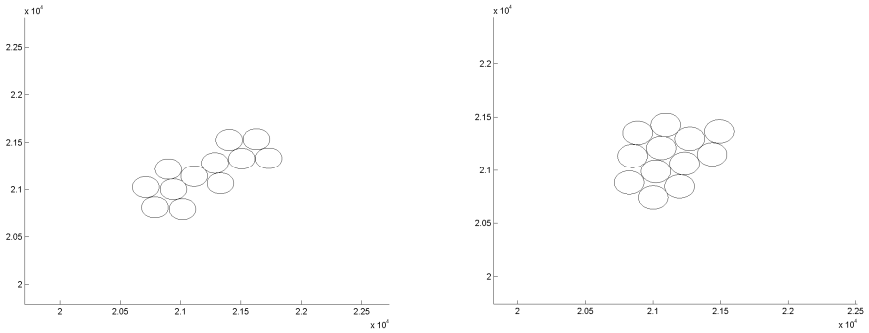
Cord-Landwehr [11], később pedig Chaudhuri [9] mutattak be olyan algoritmust, ami tetszőleges alakzatból hozta létre a kontakt gráfot, ezen kívül céljuk volt az is, hogy a létrejött alakzat a lehető legtömörebb legyen. A felhasznált modellben a robotok rendelkeztek globális navigációval és a környezetet is teljesen belátták, ami azt jelenti, hogy a többi robotot átlátszónak tekintették.

A cikkünkben bemutatásra kerülő kutatási munkánk célja egy olyan gyülekezési algoritmus kifejlesztése volt, amely eredményesen működik az Ando [1] által használt minimális tudással (limitált látótávolság, memórianélküliség, nincs kommunikáció és nem használható fel globális érzékelő rendszer) kiterjedéssel rendelkező robotok felhasználása esetén. Mivel a Czyzowicz és társai által adott gyülekezési definíció második pontja négynél nagyobb létszámú robot swarm esetén nem valósítható meg, ezért akkor tekintjük a swarm-ot összegyűltnek, ha már kialakult a kontakt gráf.

Megoldásunk alapja egy tömegközéppont (COG) alapú algoritmus, amelyet számos helyen módosítottunk annak érdekében, hogy a lehető legjobban eredményt kapjuk. Cikkünk következő két szakaszában részletesen bemutatjuk az általunk javasolt algoritmust, majd a negyedik szakaszban ismertetjük a Matlab-ban elkészített szimulációs programmal elért eredményekből levonható következtetéseket.

2. Javasolt gyülekezési algoritmus

Az általunk javasolt gyülekezési algoritmus alapvetően a lokális tömegközéppont felé való elmozduláson alapul (Center of Gravity Gathering Algorithm – COGGA). A számítások során csak a megfigyelő robot szempontjából látható robotokat vesszük figyelembe, így a láthatósági sugáron (V) kívül eső vagy takarásban levő robotok nem játszanak szerepet az algoritmus működésében. Egy robotot akkor tekintünk láthatónak, ha a középpontja a megfigyelő szemszögéből látható és V távolságon belül helyezkedik el.



1. ábra: (a) Tömegközéppont alapú (COG) gyülekezés véletlen zaj használata nélkül ($c_r=0$). (b) tömegközéppont alapú algoritmus használata zaj használatával ($c_r=0,02$).

Tapasztalatok szerint az előzőekben ismertetett megközelítés két problémát vet fel. Egyrészt amennyiben csak a lokális tömegközéppont felé való elmozdulást használjuk az algoritmusban, akkor nagyszámú robot swarm esetén a láthatósági gráf néhány lépés után megszakadhat, így a gyülekezés nem eredményes, és több magpont jöhet létre. A második probléma a gyülekezés végén létrejött kontakt gráf tömörsége. Mivel minden robot a szomszédsága tömegközéppontja felé mozdul el, a végeredmény gyakran egy elnyúlt alakzat lesz, ahogy az a *1/a* ábrán is látható.
Az előzőekben felvázolt két problémát megoldhatjuk úgy, hogy a robot által számolt célvektort eltérítjük véletlenszerűen egy előre meghatározott mértékkel. A paraméter megfelelő beállításával igen tömör alakzatot kaphatunk (ld. *1/b* ábra), így a robotok jóval kisebb helyet foglalnak a térben, mint más korábban használt algoritmusok.
Ezek alapján megadjuk a véletlennel befolyásolt célvektor számításának módját, amelyet minden robot a számítási (*Calculate*) ciklusban alkalmaz:

$$g_x = \frac{1}{n}\sum_{j=1}^{n} r_j(x)\cdot(1+c_r\cdot w_1) \tag{1}$$

$$g_y = \frac{1}{n}\sum_{j=1}^{n} r_j(y)\cdot(1+c_r\cdot w_2) \tag{2}$$

ahol n a látható robotok száma, $r_j(x)$ és $r_j(y)$ ($j=1,\ldots,n$) a j-edik látható robot x és y koordinátája, c_r az előre meghatározott véletlenszerűségi együttható, végül $w_1, w_2 \in [-0.5,0.5]$ véletlen számok (súlyok), amelyek a célvektort zajossá teszik.

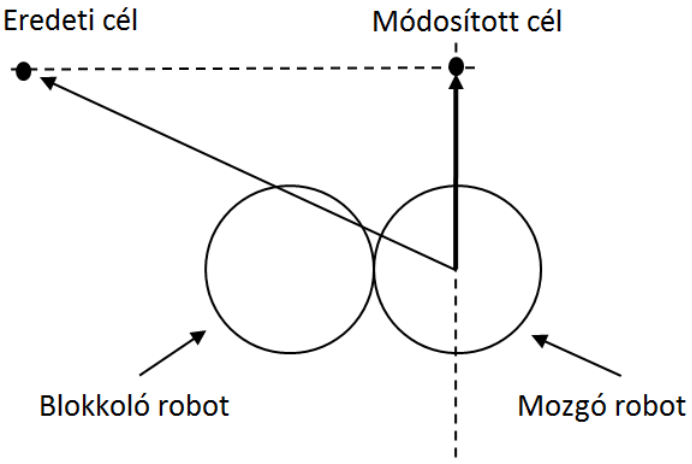
2.2. Célvektor hosszának korlátozása

Korábban már említettük, hogy önmagában a javasolt algoritmus nem ad megoldást a gyülekezés problémájára, mivel a mozgások során a láthatósági gráf megszakadhat, és nem jön létre teljesen összefüggő kontakt gráf. A gráf megszakadása ellen Ando limitáló algoritmusát használjuk [1], mely korlátozza a célvektor hosszát, annak érdekében, hogy elkerüljük a „magányossá” váló robotokat és egynél több csomópont létrejöttét.

2.3. Megoldás a blokkolás problémájára

A kiterjedéssel rendelkező robotreprezentáció egyik nagy hátránya, hogy a robotok a mozgásban blokkolhatják egymást. Pontszerű robotoknál ugyebár nincs ilyen probléma, több robot is elfoglalhat egyetlen pozíciót és nem takarja egyik a másik robotot. Tehát valamilyen megoldást kellett találni a robotok blokkolódásának elkerülésére
Egy korábbi munkánkban már foglalkoztunk kiterjedéssel rendelkező robotok gyülekezésével és ott egy lehetséges megoldást adtunk a blokkolás feloldására, amennyiben egy blokkoló robot van [15]. Ezt a megoldást bővítettük tovább, mivel a korábbi algoritmus [15] cseréjével a robotok lépései is másként alakulnak. Abban az esetben, ha a célvektor áthalad a blokkoló roboton a korábbi blokkolási algoritmus módosításával éltünk:

Ha több blokkoló robot van, **akkor**
Nem mozdulunk sehova.
Különben
Ha a kiszámított célvektor áthalad a blokkoló roboton, **akkor**
Az eredeti célvektorra merőleges irányban történjen az elmozdulás, a két lehetséges irány közül véletlenszerűen választunk. Az új mozgásvektor hosszát a következőképpen számítjuk: az eredeti célvektor hosszát egy [0,1] közötti vélten számmal szorozzuk (c_s).
Különben
Célvektorral érintő irányban mozdulunk el, ahol vektor hosszát az érintő irány vetülete adja (2. ábra).
Ha vége
Ha vége



2. ábra: Blokkolás problémája. Érintő irányba való elmozdulás egy blokkoló robot esetében.

3. Eredmények kiértékelése

Különböző gyülekezési algoritmusok használatával és paraméter beállításokkal értelemszerűen más-más kontakt gráfok alakulnak ki. Alapvetően két fontos tényezőt mérhetünk az összegyűlés beálltával: a kezdeti állapot és a kontakt gráf kialakulása között eltelt időt, valamint a kontakt gráf tömörségét, tehát hogy a robotok milyen szorosan töltik ki a rendelkezésre álló helyet. A jelen esetben a tömörségre koncentrálunk, amit többféleképpen is mérhetünk. Egyik legkézenfekvőbb lehetőség, ha vesszük a két legtávolabbi robot középpontját, majd vesszük azok távolságát. Értelemszerűen egy elnyúlt alakzatnál ez a mérőszám nagy lesz, egy kompaktabb alakzat esetén pedig kicsi. Több különböző gyülekezési algoritmus összehasonlítása esetén, ahol ez a mérőszám minimális lesz, ott található az optimális megoldás. Másik megoldás, ha számoljuk a hibát (például négyzetes hibát) az egész robot swarm súlyközéppontjától való eltéréssel, amellyel egy jól kezelhető, a gyülekezés eredményét jól jellemző mérőszámot kapunk:

$$PI = \frac{\sum_{i=1}^{N}\left(r_i(x)-m_x\right)^2 + \sum_{i=1}^{N}\left(r_i(y)-m_y\right)^2}{N \cdot H \cdot W} \cdot 1000 \tag{3}$$

$$m_x = \sum_{i=1}^{N} r_i(x) \tag{4}$$

$$m_y = \sum_{i=1}^{N} r_i(y) \tag{5}$$

ahol N a robotok száma, $r_i(x)$ és $r_i(y)$ i-dik robot x és y koordinátája a kétdimenziós Euklideszi térben. A képletben H és W a magassága, illetve a szélessége a területnek, ahol a robotok a feladatot végrehajtják. A kapott mérőszám (*PI* – Performance Indicator) segítségével már összehasonlíthatjuk az e cikkben ismertetett és a másik két korábbi gyülekezési algoritmus eredményét. Minél kisebb indikátorszámot kapunk, annál tömörebb kontakt gráf létrehozására képes az adott gyülekezési eljárás.

4. Szimulációs eredmények

A COGGA algoritmust Matlab-ban implementáltuk, majd kipróbáltuk számos szimulációs futtatás segítségével. A szimulációk során négyféle robot swarm nagysággal (N=12, 25, 50, 100) és négy kiinduló robot elrendezéssel dolgoztunk. A c_r véletlenszerű együttható esetében hat lehetséges értéket próbáltunk ki (c_r= 0; 0,005; 0,010; 0,015; 0,020; 0,025). A blokkolás feloldását megvalósító algoritmusban c_s=0,2 értékkel dolgoztunk. Ugyanazon négy robot swarm nagysággal négy kiinduló robot elrendezéssel lefuttattuk az SEC [1] és a BKF [15] algoritmusokat is. Két leállítási feltételt alkalmaztunk. Az algoritmus leállt, ha

- elérte a maximális megengedett iterációszámot (n=1000), vagy
- egy iteráció során egyetlen robotot se változtatta meg helyzetét.

Bár az összegyűléshez szükséges idő nagysága jelentősen eltért az egyes algoritmusok esetén, de mivel kutatásunk elsődleges célja a minél tömörebb végső alakzat megvalósítása volt, ezért az értékelés során csak a (3) képlettel számított mérőszámot vettük figyelembe az értékelés során. A végső összehasonlításban a négy kiinduló elrendezés mellett kapott teljesítménymutatók átlagát számítottuk. Az így kapott összesített eredmények az 1. táblázatban szerepelnek. Mindegyik kipróbált swarm nagyság esetében a c_r=0,025 paraméterrel futtatott COGGA algoritmus biztosította a legjobb eredményt.

1. táblázat. A négy robot swarm nagyság esetén tapasztalt átlagos teljesítménymutató értékek.

Method/N	12	25	50	100
SEC	0,3725	0,2441	0,6884	1,0166
BKF	0,0655	0,1743	0,3447	0,9835
COGGA – c_r=0,000	0,0887	0,7518	1,7038	5,0391
COGGA – c_r=0,005	0,1006	0,6589	0,6909	1,9295
COGGA – c_r=0,010	0,0886	0,2862	0,4757	1,3562
COGGA – c_r=0,015	0,0695	0,2446	0,3417	1,3010
COGGA – c_r=0,020	0,0602	0,1476	0,2763	0,8703
COGGA – c_r=0,025	0,0552	0,1239	0,2542	0,5490

5. Következtetések

Cikkünkben egy új robot swarm összegyűlési algoritmust mutattunk be, ami a szimulációs futtatások során megfelelő paraméterezés mellett jobb teljesítménymutatót eredményezett, mint az összehasonlítási lapként használt SEC és BKF algoritmusok. Az ismertetett modellben a robot önállóan határozza meg az elmozdulás irányát úgy, hogy az általa látott swarm tagok tömegközéppontját számítja, majd véletlenszerű nagysággal kis mértékben módosítja a célvektort. A modell blokkolás elkerülő megoldást is tartalmaz.
További kutatási célunk fuzzy logikai elemek beépíthetőségi lehetőségeinek vizsgálata az elmozdulási irány és nagyság meghatározásánál valamint a végső alakzat tömörségének értékelése során pl. összegző defuzzifikálás alkalmazásával [16].

Irodalomjegyzék

[1] Ando, H., Suzuki, I., Yamashita, M.: Formation and agreement problems for synchronous mobile robots with limited visibility. In: 1995 IEEE International Symposium on Intelligent Control, pp. 453-460. IEEE Press, New York (1995)

[2] Cieliebak, M., Flocchini, P., Prencipe, G., Santoro, N.: Solving the robots gathering problem. In: Baeten, J.C.M., Lenstra, J.K., Parrow, J., Woeginger, G.J. (eds.) ICALP 2003. LNCS, vol. 2719, pp. 1181–1196. Springer, Heidelberg (2003)

[3] R. Cohen and D. Peleg. Robot convergence via center-of-gravity algorithms. LNCS, Springer, Heidelberg, 3104: 79-88, (2004)

[4] Flocchini, P., Prencipe, G., Santoro, N., Widmayer, P.: Gathering of asynchronous robots with limited visibility. Theoretical Computer Science, vol. 337, 147–168 (2005)

[5] Cohen, R., Peleg, D.: Robot convergence via center-of-gravity algorithms. In: Kralovic, R., Sykora, O. (eds.) SIROCCO 2004. LNCS, vol. 3104, pp. 79–88. Springer, Heidelberg (2005)

[6] Souissi, S., Défago, X., Yamashita,M.: Using eventually consistent compasses to gather oblivious mobile robots with limited visibility. In: Datta, A. K., Gradinariu, M. (eds.) SSSS 2006. LNCS, vol. 4208, pp. 484–500. Springer, Heidelberg (2006)

[7] Prencipe, G.: Impossibility of gathering by a set of autonomous mobile robots. Theoretical Computer Science, vol. 384, pp. 222-231 (2007)

[8] Czyzowicz, G.J., Gasieniec, L., Pelc, A.: Gathering few fat mobile robots in the plane. Theoretical Computer Science, vol. 410, 481–499 (2009)

[9] Chaudhuri, S.G., Mukhopadhyaya, K.: Gathering asynchronous transparent fat robots. In: Janowski, T., Mohanty, H. (eds.) ICDCIT 2010. LNCS, vol. 5966, pp. 170–175.

Springer, Heidelberg (2010)
[10] B. Degener, B. Kempkes, and F.M. Heide. A local $O(n^2)$ gathering algorithm. Proceedings of the 22nd ACM Symposium on Parallelism in Algorithms and Architectures, page 224-232, (2010)
[11] Cord-Landwehr, A., Degener, B., Fischer, M., HüllmannM., Kempkes, B., Klaas, A., Kling, P., Kurras S., MärtensM., auf der Heide, F. M.,Raupach,C., Swierkot,K.,Warner D.,Weddemann, C., Wonisch, D.: Collisionless gathering of robots with an extent. In: Cerná, I., Gyimóthy, T., Hromkovic, J., Jefferey, K., Královic, R., Vukolic, M., Wolf, S. (eds.) SOFSEM 2011. LNCS, vol. 6543, pp. 178–189. Springer, Heidelberg (2011)
[12] P. Valdastri, P. Corradi, A. Menciassi, T. Schmickl, K. Crailsheim, J. Seyfried, and P. Dario. Local spread algorithms for autonomous robot systems. Communication and Swarm Intelligence Issues in a Swarm Microrobotic Platform. Robotics and Autonomous Systems, 54: 789-804 (2006)
[13] R. Cohen and D. Peleg. Local spread algorithms for autonomous robot systems. Theoretical Computer Science, 399: 71-82, (2008)
[14] S. Nouyan, A. Alexandre Campo, and M. Dorigo. Gathering path formation in a robot swarm self-organized strategies to and your way home. Swarm Intelligence, 2(1): 1-23, (2008)
[15] Bolla, K., Kovacs T., Fazekas G.: Gathering of fat robots with limited visibility and without global navigation. Swarm and Evolutionary Computation, LNCS, vol. 7269, pp. 30–38. Springer, Heidelberg (2012)
[16] Portik T., Pokorádi L.: Fuzzy Szabálybázis Alapú Kockázatértékelés Összegző Defuzzyfikáció Alkalmazásával, In: Pokorádi László (szerk.) Műszaki Tudomány az Észak Alföldi Régióban 2013, Debrecen, 2013.06.04, pp. 265-270. (ISBN:978-963-7064-30-2)

Köszönetnyilvánítás

A kutatás a TÁMOP 4.2.4.A/2-11-1-2012-0001 azonosító számú Nemzeti Kiválóság Program – Hazai hallgatói, illetve kutatói személyi támogatást biztosító rendszer kidolgozása és működtetése konvergencia program című kiemelt projekt keretében zajlott. A projekt az Európai Unió támogatásával, az Európai Szociális Alap társfinanszírozásával valósul meg.

Szerzők

Bolla Kálmán: Kecskeméti Főiskola, GAMF Kar, Informatika Tanszék. 6000 Kecskemét, Izsáki út 10. E-mail: bolla.kalman@gamf.kefo.hu

Johanyák Zsolt Csaba: Kecskeméti Főiskola, GAMF Kar, Informatika Tanszék. 6000 Kecskemét, Izsáki út 10. E-mail: johanyak.csaba@gamf.kefo.hu

Kovács Tamás: Kecskeméti Főiskola, GAMF Kar, Informatika Tanszék. 6000 Kecskemét, Izsáki út 10. E-mail: kovacs.tamas@gamf.kefo.hu

Fazekas Gábor: Debreceni Egyetem, Informatikai Kar, Információ Technológia Tanszék. 4028 Debrecen, Kassai út 26. fazekasg@inf.unideb.hu