Raport științific – proiect PN-III-P1-1.1-TE-2016-0737, nr. 32/2018

Etapa 3 – Experimente în timp real

Activitatea 3.1. Experimente pe baza soluțiilor de planificare propuse

Descrierea platformei experimentale

Experimentele în timp real s-au desfășurat într-un spațiu de lucru reprezentat de o platformă din PAL melaminat negru cu dimensiunea de 4 x 2,5 metri, platformă ce poate fi utilizată pentru cercetări privind planificarea traiectoriilor atât în domeniul de evoluție 2D cât și 3D. La nivelul platformei sunt instalate două sisteme de poziționare pentru drone:

- Microsoft Kinect v2 - senzor de mișcare compus din trei surse de iluminare IR și două camere video, una RBG și una IR. Acesta folosește principiul ToF (time-of-flight) pentru a determina distanța până la obiectele din cadru. Senzorul este montat la o înălțime de 2,8 metri, în centrul platformei și având un câmp vizual de 70x60 grade acoperă o suprafață de aproximativ 2,1 x 1,3 metri la o înălțime de 1 metru față de sol.
- Bitcraze Loco Positioning System (LPS) - sistem de poziționare bazat pe o tehnologie radio UWB (ultra wide band) ce exploatează timpul de propagare al pachetelor radio (time-of-flight) [1]. Componentele sistemului sunt poziționate la limita sau în exteriorul platformei, astfel realizându-se o acoperire completă a domeniului de evoluție 3D, așa cum este ilustrat în Fig. 2.

Sisteme pentru poziționarea absolută

Pentru urmărirea mișcării dronelor, acestea au avut montate markere color în partea superioară, pentru a facilita detecția deplasării pe axele X, Y. Detecția înălțimii de zbor a fost realizată prin intermediul unui senzor RGB-D tip Kinect v2. Acest tip de senzor prezintă următoarele caracteristici: rezoluție cameră 1920x1080 pixeli; rezoluție senzor IR 512x424 pixeli; câmp de vizualizare 70x60 grade; achiziție 30fps; rezoluție adâncime 0.5-4.5m. Senzorul RGB-D plasat deasupra platformei (Fig. 1) poate fi utilizat pentru detectarea poziției și orientării dronelor. Interacțiunea cu sistemele de calcul se poate realiza prin propriul API care poate fi apelat prin scripturi dezvoltate în diferite medii precum Matlab sau Python. Utilizarea unui senzor RGB-D are ca scop facilitarea monitorizării continue în medii indoor, simularea accesului la GPS și, în plus, posibilitatea de a calcula orientarea dronei.

În cadrul testelor realizate, folosind senzorul Kinect v2, au fost observate următoarele dezavantaje:

- Algoritmul de detecție a poziției este relativ lent (5-10 fps), rata de actualizare nefiind adecvată pentru detecția în timp real a posturii dronelor;
- Înălțimea minimă de zbor nu poate fi mai mică de 80 centimetri, deoarece caracteristicile senzorului de adâncime nu permit detectarea sub acest prag a obiectelor

de dimensiuni reduse. Astfel, scenariile de lucru nu ar fi putut include decolări și aterizări;
- Domeniul de evoluție 3D este limitat și de câmpul vizual al camerei.

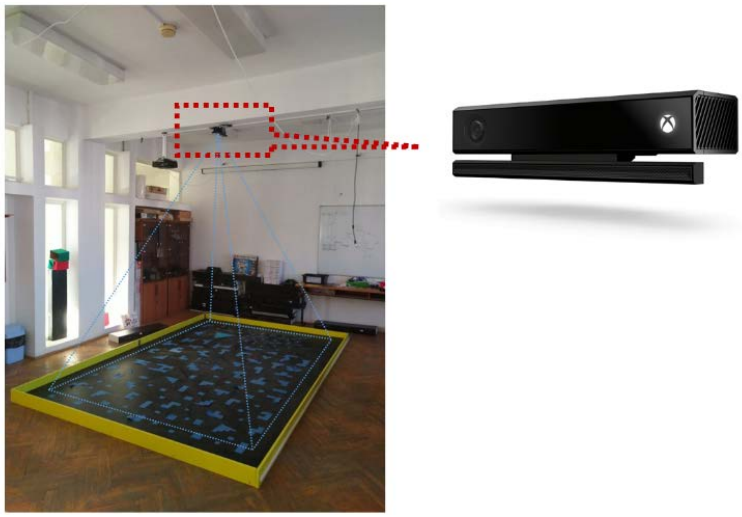

Fig. 1: Senzor RGB-D utilizat în detecția poziției și orientării dronelor în medii indoor

Sistemul LPS implementează standardul IEEE 802.15.4a-2011 (i.e. comunicații radio cu consum redus și rate de transfer mici), ce definește 16 canale UWB (Ultra-Wide Band), cu următoarele proprietăți [2]:

- canalele au o lățime de bandă de cel puțin 500 MHz și prin distribuirea uniformă a energiei în banda de emisie nu ridică probleme de coexistență cu alte tehnologii radio;
- codificările la nivel fizic presupun folosirea unor impulsuri scurte pentru transmisia datelor astfel reducându-se influența interferențelor datorate căilor de propagare multiple (eng. multipath fading);
- timpul de propagare (eng. time-of-flight) al semnalelor poate fi determinat cu o acuratețe mai mare decât în cazul tehnologiilor ce folosesc lățimi de bandă mai mici.

Tehnica de bază folosită pentru determinarea timpului de propagare nu necesită o referință de timp comună pentru sistemele care comunică. TWR (Two-way ranging) presupune generarea unei secvențe de 4 mesaje și atașarea unor mărci de timp la transmisia, respectiv recepția mesajelor. Prin măsurarea timpilor de transmisie dus-întors și a timpilor de procesare dintre recepție și transmisie, robotul care a inițiat secvența de comunicație (i.e. drona) poate extrage timpul de propagare al mesajelor, valoarea lui fiind direct proporțională cu distanța față de nodurile de referință (i.e. ancore). Prin comunicarea succesivă cu mai multe ancore și prin cunoașterea pozițiilor absolute ale ancorelor, drona poate să-și estimeze poziția prin triangularizare.

În cazul aplicațiilor complexe, existența mai multor roboți ridică probleme deosebite deoarece pentru păstrarea consistenței secvențelor de comunicație (i.e. absența întârzierilor induse) trebuie implementate mecanisme de acces la mediu de tipul CSMA/CA sau TDMA. Aceste tipuri de mecanisme au marele dezavantaj de a nu fi scalabile pentru un număr mare de noduri de comunicație - duc la creșterea numărului de coliziuni, creșterea neuniformă a întârzierilor sau la saturarea cuantelor de timp disponibile). În consecință, sistemul LPS pune la dispoziție modul de funcționare TDoA2 (Time-difference of arrival), descris de Ledergerber [2], mod ce presupune implementarea comunicației TWR între maxim 8 ancore printr-un

mecanism TDMA și recepția pachetelor TWR de un număr nelimitat de drone. Pe baza marcării timpilor de recepție a mesajelor provenite de la ancore diferite, o dronă poate să determine indirect timpii de propagare, respectiv distanțele față de ancore.

Configurația folosită este compusă din 8 ancore notate cu A0 - A7 ca în Fig. 2, dispuse la înălțimi diferite (în trei planuri), la limita sau în exteriorul platformei experimentale:

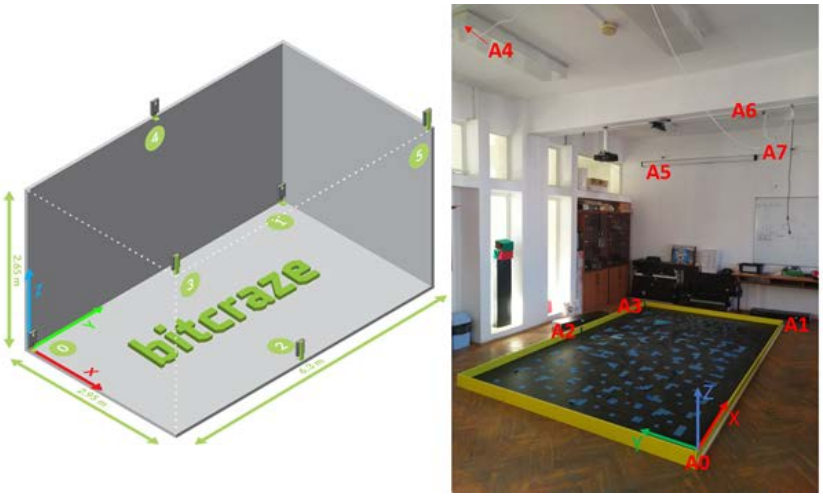

Fig. 2: Poziționarea ancorelor LPS la nivelul spațiului de lucru

În Tabelul 1 sunt ilustrate coordonatele la care sunt poziționate cele 8 ancore montate ca în Fig. 2, această configurație a ancorelor fiind aleasă în urma mai multor experimente.

Tabel 1

Ancoră	Poziție X (metri)	Poziție Y (metri)	Poziție Z (metri)
A0	0.0	0.0	0.15
A1	1.94	2.48	0.15
A2	3.91	0.0	0.15
A3	0.0	2.48	0.15
A4	-1.13	2.48	2.72
A5	1.86	0.0	2.72
A6	4.63	2.48	2.65
A7	4.63	0.0	2.65

În cadrul testelor statice, sistemul LPS are o acuratețe bună, de aproximativ 5 cm [3], însă în cadrul testelor dinamice rata de actualizare a poziției este relativ redusă și cu abateri ce pot depăși 5 centimetri, ceea ce duce la un zbor neuniform, cu oscilații (i.e. ușor observabil în cazul testelor de zbor la punct fix).

Sistem relativ de poziționare

Dronele Crazyflie 2.x pot fi echipate cu o placă de extensie (Flowdeck v2), ce poate fi utilizată pentru urmărirea mișcării. Extensia dispune de un senzor laser VL53L1X ce este utilizat pentru determinarea distanței până la sol, distanța maximă de măsurare fiind de 400 cm (performanțele scăzând în condiții de iluminare puternică). De asemenea, extensia dispune și de un senzor optic de urmărire a mișcării, PMW3901MB. Acesta este similar în construcție cu un senzor de mouse optic și poate urmări mișcarea începând cu o înălțime de 8 cm. De menționat este faptul că în foaia de catalog a senzorului nu sunt disponibile informații legate de modul în care este procesată și interpretată ieșirea, motiv pentru care producătorul dronelor Crazyflie a realizat integrarea senzorului într-o manieră experimentală, prin încercări repetate.

Testele efectuate au relevat un comportament stabil al dronei, cu ușoare oscilații doar în vecinătatea punctelor de destinație, respectiv cu o ușoară derivă în cazul zborului la punct fix. Cele două efecte se explică prin capacitatea redusă de detecție a mișcării pe axele X, Y în cazul suprafețelor colorate uniform. Stabilitatea a fost îmbunătățită prin aplicarea unor marcaje albastre la nivelul platformei (Fig. 2). Pentru acest sistem de poziționare au fost identificate următoarele neajunsuri:

- Imposibilitatea deplasării la o înălțime mai mică de 8 centimetri, datorită limitărilor constructive ale senzorului PMW3901MB;
- Drone au ca referință punctul de decolare. Astfel, în cazul scenariilor de lucru cu mai mulți roboți, sincronizarea deplasării poate fi realizată doar manual (prin cunoașterea poziției absolute a punctelor de decolare), cu riscul crescut de coliziune în cazul traiectoriilor care se intersectează datorită erorilor ce se acumulează la nivelul estimatorului de poziție;
- La deplasarea peste obstacole, dronele își modifică altitudinea de zbor. Pentru măsurarea distanței, se consideră ca referință drona și nu poate fi realizată o diferențiere între nivelul 0 al spațiului de lucru (i.e. platformă) și obiectele de pe traseu.

Configurația HW utilizată în experimentele de timp real

Pentru experimentele bazate pe soluțiile de planificare propuse a fost aleasă o soluție de poziționare mixtă, bazată pe sistemele LPS și Flowdeck. Pe lângă datele primite implicit de la unitatea inerțială, estimatorul de poziție al dronei acceptă simultan date de poziție de la cele două sisteme amintite anterior. Astfel, apar următoarele două efecte: (1) abaterea de calcul la eșantionări succesive (jitter) LPS este redusă prin influența senzorului optic prezent pe Flowdeck și (2) deriva datorată aceluiași senzor prezent pe Flowdeck este corectată periodic prin intermediul datelor de poziție de la LPS. Singurul efect nedorit rămas este cel al modificării altitudinii de zbor la deplasarea peste obiecte, însă prin fuzionarea datelor de poziție absolută și relativă se obține o amplitudine mai mică a deplasamentului pe axa Z.

Considerente de testare și calibrare

În urma testelor realizate putem să tragem următoarele concluzii referitoare la platforma experimentală bazată pe roboții Crazyflie:

- *Dronele sunt sensibile la șocurile mecanice.*

  Corpul lor este reprezentat de circuitul imprimat, la care motoarele se atașează prin intermediul unor suporturi/picioare de plastic semirigid. Suporturile și motoarele se

fixează prin presare. A fost observat faptul că în urma testelor nereușite (impact cu obiecte, pereți, podea sau aterizare bruscă de la o înălțime mare) uneori dronele își modifică comportamentul în zbor, apărând oscilații fie la decolare, fie la deplasare pe oricare din cele 3 axe. În urma impacturilor apar torsionări, deplasări, microfisuri la nivelul suporturilor ce au ca rezultat amplificarea vibrațiilor de la motoare, prin urmare suporturile trebuie înlocuite sau ajustate.

Soluție: Realizarea unor teste pentru detecția vibrațiilor înaintea fiecărui experiment. Astfel, motoarele sunt pornite/oprite în secvență și pe durata de funcționare a fiecărui motor se eșantionează accelerometrul integrat pe platformă.

- *Poziția ancorelor LPS influențează performanțele sistemului*

  În cadrul testelor inițiale de evaluare a sistemului LPS au fost observate variații de poziționare (jitter) de 30-50 cm pentru axele X, Y, respectiv 15-20 cm pentru axa Z, cu mult peste valorile specificate de producător. În urma unei analize detaliate a fost constatată influența căilor multiple de propagare ale semnalelor radio pentru unele perechi de ancore. Astfel, la nivelul secvențelor de comunicație TWR timpii de propagare ai pachetelor radio în cele două sensuri ajungeau să difere cu mai mult de 1000 cuante de timp (măsurarea distanțelor se realizează indirect pe baza unor numărătoare care contorizează durata transmisiilor și care aplică mărci de timp la nivelul pachetelor).

  Soluție: A fost dezvoltată o aplicație pentru monitorizarea duratelor de propagare ale pachetelor (Fig. 3) și s-a procedat la repoziționarea ancorelor pentru minimizarea diferențelor dintre valorile obținute pentru pachetele comunicate în cele două sensuri.

Distance avg and stdev

Received from

	A0	A1	A2	A3	A4	A5	A6	A7
A0	0.00	13921.00	14959.98	29838.64	56157.00	46075.02	62212.00	27063.00
	0.00	1381.17	6158.37	18420.69	1261.97	6735.12	1008.77	1304.70
A1	14069.00	0.00	15370.95	10104.00	39121.00	53899.00	35114.00	42941.00
	1925.35	0.00	2552.35	979.39	1133.40	1159.17	1149.63	1293.12
A2	14839.98	15297.98	0.00	37934.03	52827.00	48053.98	22365.00	28133.00
	6701.31	1880.51	0.00	13908.45	996.82	5334.75	966.15	1190.87
A3	26816.70	10120.00	55070.08	0.00	21337.00	54930.14	49147.00	7212.00
	18687.57	832.46	16248.69	0.00	1008.82	22141.06	1027.47	824.07
A4	55933.00	58529.08	52887.00	21209.00	0.00	17429.95	52424.00	55221.00
	1371.83	5330.54	969.21	1044.02	0.00	10664.80	2145.63	969.68
A5	47111.00	53945.00	56469.00	55070.14	17771.91	0.00	53167.18	47553.00
	4256.14	961.44	1382.68	22190.71	10086.82	0.00	13463.21	1484.32
A6	62238.00	35188.00	22281.00	49257.00	52376.00	56213.25	0.00	10337.00
	980.88	1054.17	1094.16	927.89	2102.95	14731.17	0.00	1436.48
A7	27063.00	42729.98	28107.00	3765.99	55177.00	47535.00	10381.00	0.00
	1261.26	4034.53	1296.60	5638.06	923.56	1470.47	1306.30	0.00

Fig. 3 (a): Valorile distanțelor după repoziționarea parțială a ancorelor

Distance avg and stdev

Received from

	A0	A1	A2	A3	A4	A5	A6	A7
A0	0.00	6781.00	59400.95	0.00	41051.07	36923.00	58016.00	19011.00
	0.00	1537.16	7652.41	0.00	6992.84	1291.50	1051.17	1247.95
A1	6757.00	0.00	6781.00	0.00	56171.00	48935.00	29718.00	46775.00
	1352.19	0.00	1716.39	0.00	1464.26	1004.53	1024.34	1051.04
A2	60103.00	6941.00	0.00	0.00	53103.96	53505.00	19865.00	28155.00
	1546.87	1166.89	0.00	0.00	4822.74	1131.72	1153.67	1083.71
A3								
A4	39323.10	55877.00	53433.00	0.00	0.00	14773.78	53582.00	56821.00
	9928.01	1250.74	981.47	0.00	0.00	12050.70	1204.33	983.81
A5	36963.00	49063.00	53371.00	0.00	13555.69	0.00	44941.45	48385.00
	1169.12	1173.24	1194.69	0.00	11895.20	0.00	20242.28	1175.55
A6	58146.00	29622.00	19935.00	0.00	53498.00	48795.30	0.00	11385.00
	1065.01	889.75	1121.92	0.00	1338.89	17991.19	0.00	1009.49
A7	19133.00	46837.00	28089.00	0.00	56757.00	48319.00	11433.00	0.00
	1233.48	1069.27	860.62	0.00	1078.99	1109.84	992.83	0.00

Fig. 3 (b): Valorile distanțelor după repoziționarea ancorelor

Algoritmi suplimentari și exemple

Evitarea coliziunilor în etapa de planificare

Soluțiile de planificare propuse anterior returnează secvențe de mișcare și de efectuare a acțiunilor de către roboții echipei, împreună cu momente necesare de sincronizare, acestea fiind prezentate pe larg în monografia [4]. În timpul mișcării trebuie evitate coliziunile între roboți, pentru acest lucru putându-se apela la procedee relaționate cu domeniul de alocare a resurselor. Ca noutate, cercetările raportate în lucrarea [5] încorporează evitarea coliziunilor direct în problema de planificare pe baza unei specificații Booleene. Astfel, sunt extinse soluțiile anterioare din [6], unde nu se putea garanta evitarea coliziunilor, ci doar se încerca minimizarea posibilității de apariție a acestora.

Pe scurt, se presupune o echipă cu N_r roboți, un set de propoziții atomice (acțiuni și regiuni) $Y = \{y_1, y_2 \dots, y_{|Y|}\}$ și o specificație Booleană pentru întreaga echipă. Specificația este impune cerințe intermediare (de-a lungul traiectoriilor) folosind setul $Y_i = \{Y_1, Y_2 \dots, Y_{|Y|}\}$ și cerințe pentru starea finală a agenților folosind setul $Y_f = Y$. Formula Booleană este presupusă de formă normal conjunctivă, precum $\varphi = \varphi_1 \wedge \dots \varphi_n$, fiecare termen φ_i fiind o disjuncție cu elemente din setul Y_i sau Y_f. Pentru echipa robotică se consideră un model RMPN (Robot Motion Petri Net) notat cu Q, definit în studii anterioare precum [6-7]. Pentru formularea unei probleme de planificare care să garanteze lipsa coliziunilor, rezultatele din [5] necesită următoarele ipoteze adiționale față de [6]:

(i) Fiecare disjuncție φ_i conține elemente fie din setul Y_i, fie din Y_f, dar nu din ambele.

(ii) Partea formulei referitoare la specificații intermediare poate fi satisfăcută de o singură configurație a celor N_r roboți, adică există cel puțin un marcaj al modelului Q care îndeplinește cerințele intermediare. Bineînțeles, acest lucru este adevărat pentru cerințele în starea finală, deoarece altfel formula nu ar putea fi îndeplinită.

(iii) Fiecare disjuncție cu elemente din Y_i (cerințe intermediare) poate conține fie mai multe elemente ne-negate, fie este egală doar cu un singur element negat.

(iv) La momentul inițial roboții sunt plasați în celule diferite din partiția P, adică marcajul inițial al rețelei Petri Q satisface $m_0[p] \leq 1, \forall p \in P$.

Se observă că presupunerile (i)-(iii) implică misiuni Booleene mai restrictive decât în [6]. De exemplu, din cauza ipotezei (i) nu putem avea formule conținând $(Y_1 \vee y_2)$, iar în cazul unui robot ipoteza (ii) implică imposibilitatea cerinței $(Y_1 \wedge Y_2)$, pe când soluția din [6] ar fi implicat un plan în care robotul satisfăcea pe rând Y_1, respectiv Y_2. Însă, precum a fost menționat, aceste ipoteze permit o soluție de planificare care garantează evitarea coliziunilor, chiar dacă traiectoriile se intersectează.

Metoda propusă în [5] se bazează pe rezolvarea a două probleme de programare liniară cu necunoscute întregi și reale (MILP), una pentru cerințele intermediare, iar cealaltă pentru cerințele finale. Aceste probleme sunt prezentate în Fig. 4, iar pentru detalierea notațiilor aferente direcționăm cititorul spre studiile [4-5].

$$\min \mathbf{1}^T \cdot \sum_{j=1}^{N_r+1} j \cdot \boldsymbol{\sigma}_j$$
$$\text{s.t. } \boldsymbol{m}_j = \boldsymbol{m}_{j-1} + \boldsymbol{C} \cdot \boldsymbol{\sigma}_j,\ j = 1, 2, \ldots, N_r + 1$$
$$\sum_{\gamma \in \mathcal{Y}_i \setminus \bar{\mathcal{Y}}} (\alpha_j(\gamma) \cdot x_\gamma) \geq 1 + \sum_{\gamma \in \mathcal{Y}_i \setminus \bar{\mathcal{Y}}} min\left(\alpha_j(\gamma), 0\right), \forall \varphi_j$$
$$N_r \cdot x_\gamma \geq \boldsymbol{v}_\gamma \cdot \boldsymbol{m}_{N_r+1}, \forall \gamma \in \mathcal{Y}_i \setminus \bar{\mathcal{Y}}$$
$$x_\gamma \leq \boldsymbol{v}_\gamma \cdot \boldsymbol{m}_{N_r+1}, \forall \gamma \in \mathcal{Y}_i \setminus \bar{\mathcal{Y}}$$
$$\boldsymbol{\eta} \cdot \boldsymbol{\sigma}_j = 0,\ j = 1, 2, \ldots, N_r + 1$$
$$\boldsymbol{Post} \cdot \boldsymbol{\sigma}_j + \boldsymbol{m}_{j-1} \leq \mathbf{1},\ j = 1, 2, \ldots, N_r + 1$$
$$\boldsymbol{m}_j \in \mathbb{R}_{\geq 0}^{|P|},\ j = 1, 2, \ldots, N_r + 1,$$
$$\boldsymbol{\sigma}_j \in \mathbb{N}_{\geq 0}^{|T|},\ j = 1, 2, \ldots, N_r + 1, \boldsymbol{x} \in \{0, 1\}^{|\mathcal{Y}_i \setminus \bar{\mathcal{Y}}|}.$$

(a)

$$\min \mathbf{1}^T \cdot \sum_{j=1}^{N_r+2} j \cdot \boldsymbol{\sigma}_j$$
$$\text{s.t. } \boldsymbol{m}_j = \boldsymbol{m}_{j-1} + \boldsymbol{C} \cdot \boldsymbol{\sigma}_j, j = 1, 2, \ldots, N_r + 2,$$
$$\sum_{\gamma \in \mathcal{Y}_f} (\alpha_j(\gamma) \cdot x_\gamma) \geq 1 + \sum_{\gamma \in \mathcal{Y}_f} min\left(\alpha_j(\gamma), 0\right), \forall \varphi_j$$
$$N_r \cdot x_\gamma \geq \boldsymbol{v}_\gamma \cdot \boldsymbol{m}_{N_r+2}, \forall \gamma \in \mathcal{Y}_f$$
$$x_\gamma \leq \boldsymbol{v}_\gamma \cdot \boldsymbol{m}_{N_r+2}, \forall \gamma \in \mathcal{Y}_f$$
$$\boldsymbol{\eta} \cdot \boldsymbol{\sigma}_j = 0,\ j = 1, 2, \ldots, N_r + 1$$
$$\boldsymbol{Post} \cdot \boldsymbol{\sigma}_j + \boldsymbol{m}_{j-1} \leq \mathbf{1},\ j = 1, 2, \ldots, N_r + 2$$
$$\boldsymbol{m}_{N_r+1} - \boldsymbol{Pre} \cdot \boldsymbol{\sigma}_{N_r+2} \geq \mathbf{0}$$
$$\boldsymbol{m}_j \in \mathbb{R}_{\geq 0}^{|P|},\ j = 1, 2, \ldots, N_r + 2,$$
$$\boldsymbol{\sigma}_j \in \mathbb{N}_{\geq 0}^{|T|},\ j = 1, 2, \ldots, N_r + 2, \boldsymbol{x} \in \{0, 1\}^{|\mathcal{Y}_f|}.$$

(b)

Fig. 4: Formulări MILP asigurând evitarea coliziunilor, corespunzătoare cerințelor Booleene: (a) intermediare, (b) finale.

Pe scurt, problema MILP (a) generează strategii robotice la sfârșitul cărora roboții satisfac cerințele intermediare, iar MILP (b) generează continuări către pozițiile finale. Sunt necesare sincronizări în marcajele intermediare ale celor două probleme MILP pentru a se asigura evitarea coliziunilor. Funcțiile de cost sunt gândite astfel încât (atunci când este posibil) unele marcaje succesive să rezulte identice, reducând astfel momente de sincronizare.

Pseudo-codul soluției de planificare Booleană cu evitarea coliziunilor este dat în Algoritmul 1 de mai jos. Se pornește de la testarea valorii de adevăr a ipotezelor (i)-(iv) – în caz negativ se utilizează metoda din [6] (care nu garantează evitarea coliziunilor), iar în caz afirmativ se continuă cu soluțiile corespunzătoare MILP (a) și (b), asigurându-se sincronizările necesare.

Algorithm 1. Solution pseudo-code
Input: RMPN $\mathcal{Q}$, set $\mathcal{Y}$, formula φ, N_r
Output: Robot movement strategies
  if *Any assumption (i)-(iv) is False* **then**
    Use the more complex and general method from [6] and Return solution;
  Solve MILP (a);
  Construct movement plans from non-empty $\boldsymbol{\sigma}_j$, $j = 1, \ldots, N_r + 1$;
  Robots synchronize in each marking $\boldsymbol{m}_j$ for which $\boldsymbol{\sigma}_j \neq \mathbf{0}$, $j = 1, \ldots, N_r + 1$;
  Set $\boldsymbol{m}_0 := \boldsymbol{m}_{N_r+1}$, to use in MILP (b);
  Solve MILP (b);
  Robots further move based on non-empty $\boldsymbol{\sigma}_j$, $j = 1, \ldots, N_r + 2$;
  Robots synchronize in each marking $\boldsymbol{m}_j$ for which $\boldsymbol{\sigma}_j \neq \mathbf{0}$, $j = 1, \ldots, N_r + 2$

Formulările MILP (a) și (b) oferă soluții complete pentru problema de planificare. Din punct de vedere al complexității (judecată prin numărul necunoscutelor), problema (a) are $(N_r + 1) \cdot |P|$ variabile reale, $(N_r + 1) \cdot |T|$ variabile întregi și $|Y \setminus \underline{Y}|$ variabile binare, unde $|P|$ și $|T|$ reprezintă numărul de poziții, respectiv tranziții ale modelului Q, iar $|Y \setminus \underline{Y}|$ este numărul de propoziții atomice care apar fără a fi negate în specificația φ. Problema (b) are $(N_r + 2) \cdot |P|$ variabile reale, $(N_r + 2) \cdot |T|$ variabile întregi și $|Y|$ variabile binare. În contrast, soluția din [6] avea un număr de variabile dependent de un parametru k ales de utilizator, valori

prea mici ale acestui parametru putând duce la infezabilitatea problemei de optimizare, iar valori mari crescând complexitatea optimizării.

Pentru exemplificarea procedurii de planificare de mai sus, considerăm scenariul din Fig. 5 și specificația $\varphi = (\Lambda_{i=1}^{10} \neg Y_i) \Lambda (\Lambda_{i=11}^{20} y_i)$, care impune atingerea în starea finală a regiunilor din partea dreaptă, evitând de-a lungul traiectoriilor regiunile din mijloc.

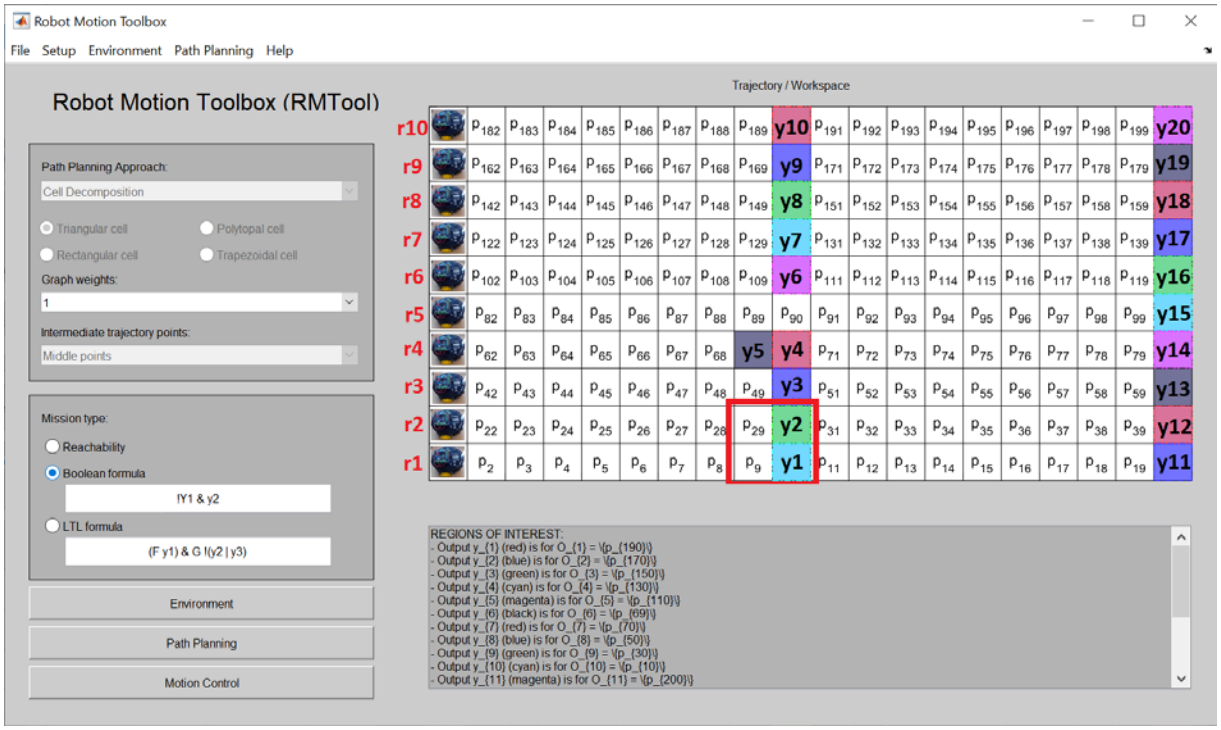

Fig. 5: Domeniu de evoluție cu 10 roboți, 200 de celule în partiție și 20 de regiuni de interes.

Implementarea a fost realizată în Matlab și integrată în RMTool (Robot Motion Toolbox) [7]. O soluție conform Algoritmului 1 a fost obținută în aproape o secundă (pe un calculator cu procesor i7 gen. 8), problemele de optimizare având (a) 10360 variabile, respectiv (b) 11300 variabile. Coliziunile sunt evitate, în acest sens fiind necesare 10 momente de sincronizare, ilustrate în Fig. 6 împreună cu traiectoriilor robotice obținute. Dacă s-ar fi utilizat metoda din [6], ar fi fost posibilă apariția coliziunilor (deoarece unele marcaje intermediare ale modelului Q ar fi avut mai mult de două jetoane în aceeași poziție / celulă a partiției), pe când acest lucru nu este posibil folosind metoda menționată anterior [5]. Devine clară necesitatea sincronizărilor între roboți, acest lucru fiind transpus în practică prin experimente cu drone, prezentate spre sfârșitul documentului curent.

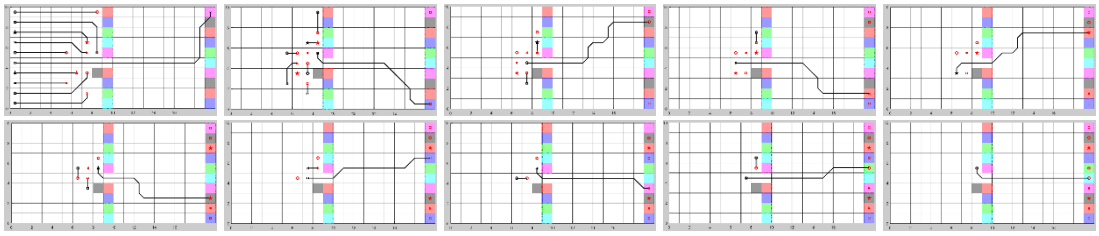

Fig. 6: Momente de sincronizare necesare evitării coliziunilor robotice.
Traiectoriile obținute satisfac specificația φ.

Pentru a extinde sfera aplicabilității specificațiilor de nivel înalt, lucrarea [8] consideră o specificație Booleană globală doar pentru stări finale, dar într-un scenariu distribuit, unde roboții pot comunica între ei doar într-o anumită rază. În plus, roboții au doar informații parțiale asupra domeniului de evoluție și estimează zonele de interes din vecinătatea lor în timpul mișcării. Fiecare robot rulează un algoritm local prin care își actualizează estimările pe baza comunicațiilor cu roboții apropiați și încearcă satisfacerea specificației globale prin rezolvarea unei instanțe locale a unei probleme de optimizare.

Problemă de planificare pentru livrarea unor resurse

Un alt scenariu exemplificat pe o platforma de timp real se referă la echipe de drone care au acțiuni de preluare și livrare bunuri în regiuni de interes având restricții de energie [9]. Pentru acest scenariu se consideră următoarele ipoteze de lucru:

I. Se consideră un spațiu indoor în care sunt poziționate un număr finit de depozite de stocare bunuri (Fig. 7);
II. Se consideră un set de drone cu număr egal cu cel al depozitelor de bunuri;
III. Fiecare dronă poate transporta un singur tip de bun, un număr egal cu *c* din aceste bunuri, are energie egală cu *E* și se poate reîncărca în depozitele de stocare.
IV. În spațiul de lucru există un număr finit de centre de colectare bunuri, fiecare centru putând accepta un număr predefinit din fiecare categorie de bunuri.

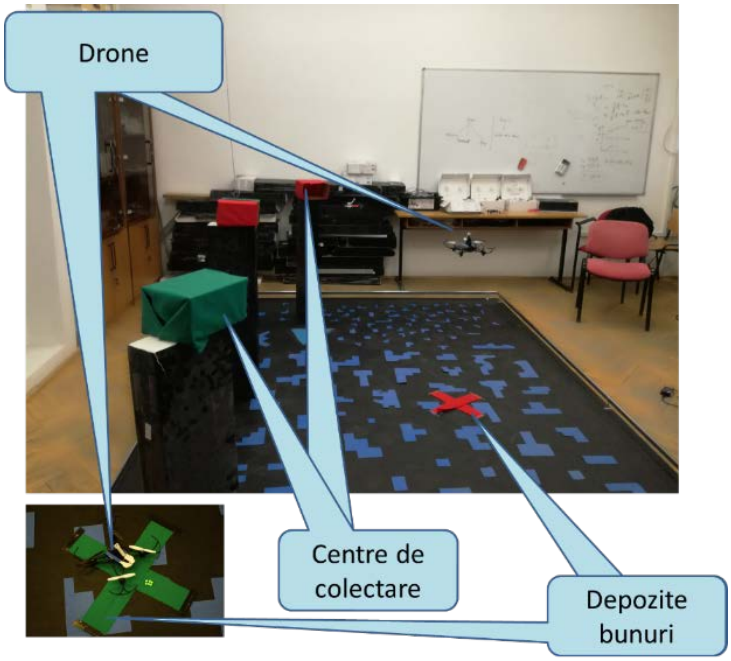

Fig. 7: Spațiul de lucru în platforma experimentală de timp real

Problema care modelează scenariul descris are ca scop calcularea secvențelor de preluare și transport de bunuri pentru fiecare dronă, optimizând consumul de energie. Adițional, toate transporturile trebuie realizate într-o durată maximă egală cu T. Soluția constă în construirea de formulări BIP pentru fiecare dronă (Fig. 8). Depozitul de bunuri și numărul de bunuri pe care trebuie să le livreze $|G|$ sunt elemente ale mulțimii $P = \{0, 1, 2, \ldots, |G|\}$.

Pentru realizarea zborului și finalizarea acțiunilor de livrare cu drone se consumă unități de energie și timp. Drona poate realiza maxim N_t tururi pentru a realiza toate livrările.

$$(i) \quad \min_{x_g^t, y_{i,j}^t, z^t} \sum_{t\in\{1,2,\ldots,N_t\}} \Big(\sum_{g\in G} u^e \cdot x_g^t + \sum_{i,j\in P,\, i\neq j} m^e \cdot d(i,j) \cdot y_{i,j}^t \Big)$$

subject to:

$$(ii) \quad \sum_{g\in G} u^e \cdot x_g^t + \sum_{i,j\in P,\, i\neq j} m^e \cdot d(i,j) \cdot y_{i,j}^t \leq E\,,\ \forall t \in \{1,2,\ldots,N_t\}$$

$$(iii) \quad \sum_{t\in\{1,2,\ldots,N_t\}} \Big(\sum_{g\in G} u^\tau \cdot x_g^t + \sum_{i,j\in P,\, i\neq j} m^\tau \cdot d(i,j) \cdot y_{i,j}^t + f^\tau \cdot z^t \Big) \leq T$$

$$(iv) \quad \sum_{g\in G} x_g^t \leq c\,,\ \forall t \in \{1,2,\ldots,N_t\}$$

$$(v) \quad \sum_{t\in\{1,2,\ldots,N_t\}} x_g^t = 1\,,\ \forall g \in G$$

$$(vi) \quad \sum_{g\in G} x_g^t \geq z^t\,,\ \forall t \in \{1,2,\ldots,N_t\}$$

$$(vii) \quad \sum_{g\in G} x_g^t \leq c \cdot z^t\,,\ \forall t \in \{1,2,\ldots,N_t\}$$

$$(viii) \quad \sum_{i\in P,\, i\neq g} y_{i,g}^t = x_g^t\,,\ \forall t \in \{1,2,\ldots,N_t\},\ \forall g \in G$$

$$(ix) \quad \sum_{j\in P,\, j\neq g} y_{g,j}^t = x_g^t\,,\ \forall t \in \{1,2,\ldots,N_t\},\ \forall g \in G$$

$$(x) \quad \sum_{g\in G} y_{0,g}^t = z^t\,,\ \forall t \in \{1,2,\ldots,N_t\}$$

$$(xi) \quad \sum_{g\in G} y_{g,0}^t = z^t\,,\ \forall t \in \{1,2,\ldots,N_t\}$$

$$(xii) \quad \sum_{\alpha\in P\setminus M} \sum_{\beta\in M} y_{\alpha,\beta}^t \geq x_g^t\,,\ \forall t \in \{1,2,\ldots,N_t\},\ \forall M \subset G,\ \forall g \in M$$

$$(xiii) \quad x_g^t, y_{i,j}^t, z^t \in \{0,1\}\,,\ \forall g \in G,\ \forall i,j \in P\,(i \neq j),\ \forall t \in \{1,2,\ldots,N_t\}$$

Fig. 8: Formulare BIP scenariu individual drona

Validarea soluției propuse a fost realizată în două faze. Prima fază folosind un simulator, în timp ce în a doua fază soluția a fost transpusă într-o aplicație de timp real. Pentru simulare (Fig. 9) se consideră un spațiu paralelipipedic în care evoluează două drone (una verde una roșie) având capacitate c=3. Depozitele de bunuri sunt considerate la nivelul solului (înălțime 0) și centrele de colectare sunt la o înălțime egală cu 3 unități de măsură. Formularea BIP pentru drona verde are 72 de necunoscute si 188 de restricții, în timp ce pentru drona roșie sunt 16 necunoscute și 25 de restricții. Formulările BIP au fost generate automat și rezolvate în mai puțin de 0.15 secunde folosind mediile Matlab și CPLEX.

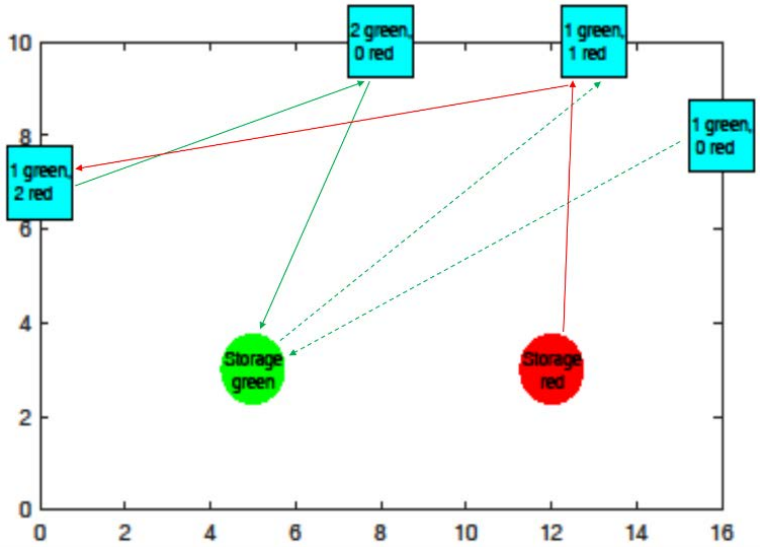

Fig. 9 Simularea soluției propuse

După validarea prin simulare a fost considerată o aplicație de timp real. Două drone sunt utilizate pentru a îndeplini sarcinile de transportare a bunurilor către centrele de colectare în spațiul descris în Fig. 10. Planificarea traiectoriilor este realizată off-line în mediul Matlab după o identificare prealabilă a dronelor, centrelor de colectare și a depozitelor. Identificarea se realizează folosind informații de culoare și adâncime achiziționate de un sensor RGB-D tip kinect v2 și procesate cu OpenCV. După construirea traiectoriilor ce implică și acțiuni de preluare și depunere bunuri, acestea sunt transmise via bluetooth dronelor utilizând un script Python.

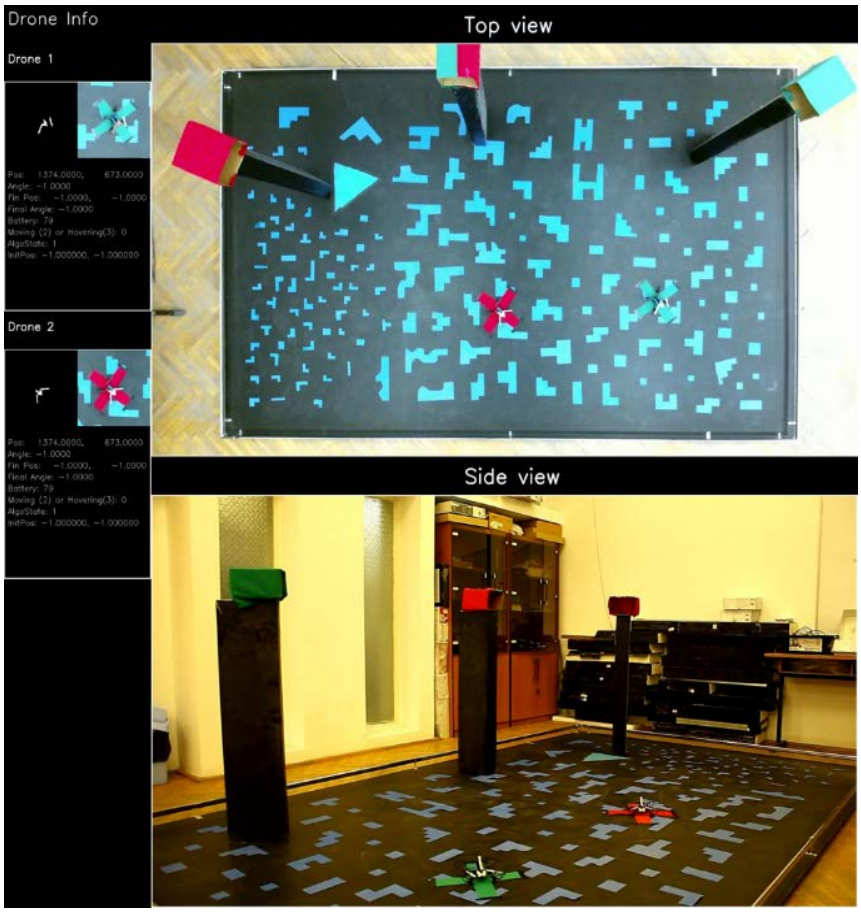

Fig. 10: Platformă experimentală

Planificarea traiectoriilor asigură realizarea sarcinilor cu un consum minim de energie și într-un timp sub 10 secunde. Experimentele în timp real pot fi vizualizate accesând link-ul: https://www.youtube.com/watch?v=SDDywcx6rms. Scenariul anterior poate fi extins de exemplu inserând pauze de alimentare cu energie a dronelor prin aterizarea pe stații de încărcare wireless, în funcție de energia curentă și cea necesară pentru preluările și livrările asignate.

Simulare soluții de planificare cu specificații bazate pe acțiuni

Exemplul 1. Se considera o echipa de roboti $R = \{r_1, r_2\}$, cu capabilitati identice, prezenti intr-un mediu industrial automatizat reprezentat de mediul de lucru rectangular din Fig. 11. Mediul de lucru consta din trei zone de interes: zona de productie (marcata cu rosu), zona depozit materiale (marcata cu albastru), zona de mentenanta echipamente (marcata cu verde).

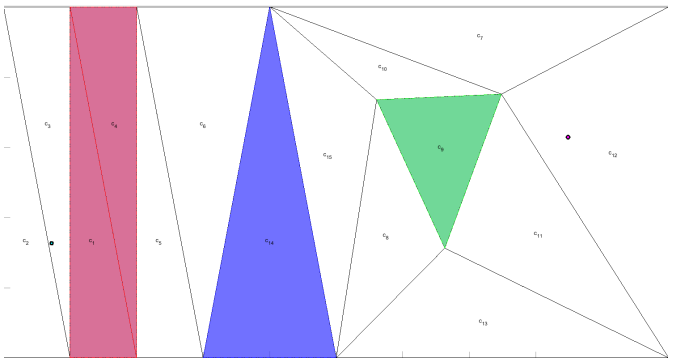

Fig. 11: Harta descompunerii pe regiuni a mediului de lucru in exemplul 1.

Mediul de lucru este descompus in celule obtinandu-se setul $C = \{c_i, i \in [1,15]\}$, si presupunem ca robotii sunt plasati in starea initiala in regiunile c_3 si c_{12}.

Se considera actiunea π_1 de "verificare proces productie", actiunea π_2 de "verificare a stocului de materiale" si actiunea π_3 de "reparare echipament" astfel incat setul $\Pi = \{\pi_1, \pi_2, \pi_3\}$.

Actiunea π_1 se poate efectua in regiunile c_1 si c_4 corespunzand zonei de productie, actiunea π_2 in c_{14} repezentand zona de depozitare, si actiunea π_3 in c_9 corespunzatoare zonei de mentenanta, astfel ca setul C^* al regiunilor de interes este definit ca $C^* = c_1, c_3, c_4, c_9, c_{12}, c_{14}$, incluzand si regiunile in care sunt plasati initial robotii.

Consideram o specificatie pentru echipa de roboti mobili, de forma "Verifica proces productie, verifica stoc materiale si asigura repararea echipamentelor defecte". Formal, specificatia LTL va avea forma $\phi_1 = (F\pi_1)\&(F\pi_2)\&(F\pi_3)$. Aceasta specificatie impune realizarea cooperativa de catre echipa de roboti a actiunilor specificate in regiunile in care acestea se pot efectua.

Simularea acestei specificatii in RMTool [4] furnizeaza solutia din Fig. 12. Traiectoriile robotilor sunt reprezentate prin conectarea punctelor mediane ale segmentelor partajate de celule adiacente.

Obtinerea acestei solutii folosind metoda bazata pe retele Petri descrisa in raportul etapei anterioare de proiect presupune definirea setului de pozitii si de tranzitii ale retelei pe baza ipotezelor de lucru si reducerea grafului de adiacenta corespunzator celulelor mediului de lucru la graful de regiuni de interes, retinand drumul cel mai scurt in graful complet intre oricare doua noduri din graful redus. Figura de mai sus ilustreaza traiectoriile robotilor corespunzatoare detalierii acestor drumuri.

Ipotezele de lucru conduc la definirea setului P de pozitii in reteaua Petri prin $P = \{p_i, i \in [1,6]\}$.

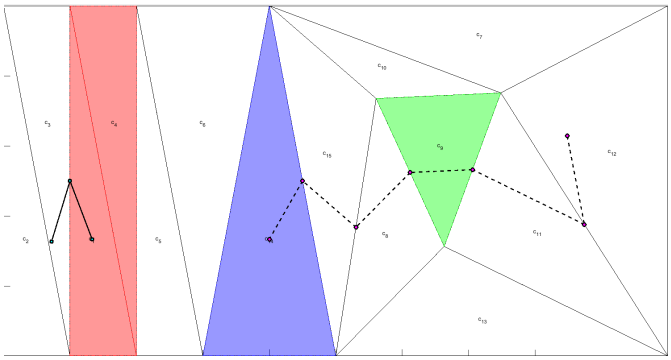

Fig. 12: Traiectoriile ce satisfac specificatia ϕ_1 din exemplul 1.

$$\begin{array}{l|l} p_1 : (\pi_1, c_1) & p_4 : (\pi_3, c_9) \\ p_2 : (\pi_1, c_4) & p_5 : (\emptyset, c_3) \\ p_3 : (\pi_2, c_{14}) & p_6 : (\emptyset, c_{12}) \end{array}$$

Setul de tranzitii este reprezentat de $T = \{t_1, \ldots t_{20}\}$, ce contine toate tranzitiile intre oricare doua pozitii distincte din setul P, mai putin cele inspre p_5 si p_6 ce au asociata actiunea $\emptyset$. Spre exemplu, tranzitia intre p_1 si p_3 are semnificatia deplasarii unui robot intre celulele c_1 si c_{14} si efectuarea in c_{14} a actiunii π_2. Tranzitia inversa, intre p_3 si p_1 semnifica deplasarea unui robot intre c_{14} si c_1 si efectuarea actiunii π_1 in celula c_1. Tranzitia intre p_1 si p_4 echivaleaza cu deplasarea unui robot intre c_1 si c_9, pe drumul cel mai scurt, si efectuarea π_3 in celula c_9.

Marcajul initial al sistemului este $m_0 = [0,0,0,0,1,1]^T$, alfabetul de iesire este $\Pi = \{\pi_1, \pi_2, \pi_3\}$ iar functia observatiilor: $h(p_1) = h(p_2) = \{\pi_1\}$, $h(p_3) = \{\pi_2\}$, $h(p_4) = \{\pi_3\}$, $h(p_5) = h(p_6) = \emptyset$.

Vectorul caracteristic al lui π_1 este $v_1 = [1,1,0,0,0,0]$ avand in vedere ca iesirea π_1 poate fi observata in p_1 si p_2, $v_2 = [0,0,1,0,0,0]$, iar $v_3 = [0,0,0,1,0,0]$.

Astfel, $V = \begin{bmatrix} 1 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \end{bmatrix}$.

Deoarece $V \cdot m_0 = [0,0,0]^T$, nicio observatie din Π nu este activa in m_0.

Exemplul 2. In acest exemplu vom ilustra situatia in care se doreste combinarea unei specificatii pe actiuni cu specificatii de miscare pentru robotii mobili. Consideram mediul de lucru si setul de roboti din exemplul 1 si adaugam in misiunea echipei constrangerea evitarii unei regiuni, de exemplu: "Verifica stocul de materiale din depozit si efectueaza reparatiile echipamentelor defecte, evitand trecerea prin zona de productie". Pentru a formaliza evitarea unei regiuni in cadrul unei specificatii pe actiuni vom defini o noua actiune de tipul "evita regiune". Vom obtine astfel setul de actiuni $\Pi = \{\pi_1, \pi_2, \pi_3\, \pi_4\}$, unde π_4 corespunde actiunii de evitare a unei regiuni, iar restul actiunilor sunt cele din exemplul 1:

$$
\begin{array}{l|l}
p_1 : (\pi_1, c_1) & p_5 : (\pi_4, c_1) \\
p_2 : (\pi_1, c_4) & p_6 : (\pi_4, c_4) \\
p_3 : (\pi_2, c_{14}) & p_7 : (\emptyset, c_3) \\
p_4 : (\pi_3, c_9) & p_8 : (\emptyset, c_{12})
\end{array}
$$

iar specificatia LTL va fi formulata sub forma $\phi_2 = (F\pi_2)\&(F\pi_3)\&G!(\pi_4)$. Simularea misiunii in RMTool furnizeaza solutia din Fig. 13.

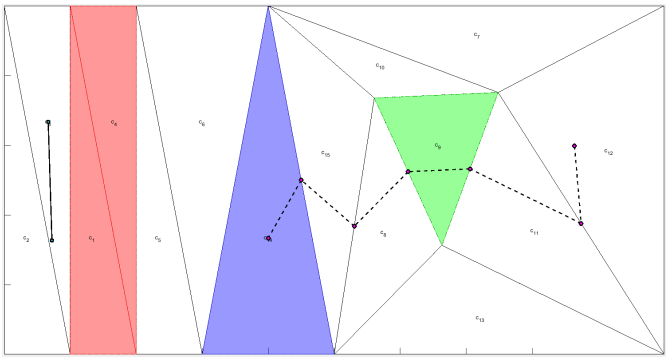

Fig. 13: Traiectoriile ce satisfac specificatia ϕ_2 din exemplul 2. Robotul r_1 nu participa la indeplinirea misiunii datorita restrictiei de vizitare a regiunii de productie (rosu).

Se observa ca intreaga misiune este satisfacuta cu un singur robot, avand in vedere ca robotul r_1 nu poate contribui datorita restrictiei de vizitare a zonei de productie.

Daca se elimina din misiune cerinta de reparare a echipamentelor defecte iar restrictia de vizitare se introduce si asupra zonei de mentenanta, misiunea poate fi specificata formal prin $\phi_3 = (F\pi_2)\&G!(\pi_4)$ avand solutia ilustrata in Fig. 14.

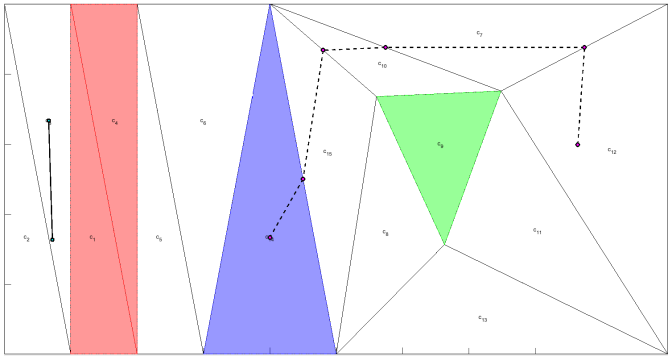

Fig. 14: Traiectorii ce satisfac specificatia ϕ_3. Robotul r_1 nu participa la indeplinirea misiunii datorita restrictiei de vizitare a regiunii de productie (rosu), iar robotul r_2 indeplineste misiunea de verificare a stocului de materiale evitand trecerea prin zona de mentenanta.

Pentru aceasta configuratie, setul de pozitii in reteaua Petri este definit de:

$$
\begin{array}{l|l}
p_1 : (\pi_1, c_1) & p_6 : (\pi_4, c_4) \\
p_2 : (\pi_1, c_4) & p_7 : (\pi_4, c_9) \\
p_3 : (\pi_2, c_{14}) & p_8 : (\emptyset, c_3) \\
p_4 : (\pi_3, c_9) & p_9 : (\emptyset, c_{12}) \\
p_5 : (\pi_4, c_1) &
\end{array}
$$

Descrierea experimentelor adiționale și analiza performanțelor

În cadrul experimentelor adiționale s-au realizat teste în timp real cu mai multe drone Crazyflie 2.0, folosind configurația hardware descrisă în prima secțiune a documentului. În acest context, s-au realizat teste experimentale în timp real cu 3 și 4 drone efectuând diferite acțiuni sincronizate. Avantajele oferite de configurația mixtă LPS plus Flowdeck pot fi evidențiate în problemele de planificare în care sunt implicate mai multe drone cu sarcini diferite.

În etapa anterioară a acestui proiect, s-au realizat teste cu o singură dronă CF2, considerând configurații diferite pentru sistemele de poziționare: *Drona CF2 (fără sistem de poziționare), Drona CF2 și Z-Ranger (feedback poziție doar pe axa Z)* și *Drona CF2 și Flow Deck v2,* fiind raportate cele mai bune rezultate [10]. Pentru fiecare configurație, au fost considerate 4 scenarii de testare. Experimentele s-au desfășurat impunând o înălțime minimă constantă de 1 m față de sol, restricție motivată de precizia senzorului Kinect, care a fost folosit pentru evaluarea preciziei estimatorului de poziție. Datorită acestui neajuns, experimentele din această a III-a etapă care au implicat acțiuni de decolare și aterizare a dronelor Crazyflie, nu au putut fi validate prin folosirea senzorului Kinect.

Experimentele în timp real au implicat câteva scenarii de testare, cu 2 echipe a câte 2 drone (Fig. 15) și 3 drone (Fig. 16) ce realizează acțiuni cu unul sau mai multe puncte de sincronizare. În Tabel 2 sunt prezentate atât scenariile de test rulate în care echipele de drone colaborează la îndeplinirea unei specificații globale, cât și scenariile cu evitarea de obstacole. Traiectoriile pot fi planificate pe baza algoritmilor dezvoltați, iar momentele de sincronizare returnate asigură evitarea coliziunilor.

Tabel 2. Scenarii de testare

Scenarii de test	**Descriere acțiuni (decolare, survolare/evitare regiuni/obstacole și aterizare)**
1.	Două echipe a câte 2 drone (R1, R2) și (R3, R4) poziționate la puncte fixe (rezultate din planificare).
2.	Decolarea simultană a celor 4 drone la punct fix la înălțimea de 2 metri timp de 5 secunde.
3.	Coborâre simultană a unei echipe de 2 drone la înălțimea de 1 metru (R1, R2), cealaltă echipă de 2 drone (R3, R4) menținându-și zborul la punct fix la înălțimea de 2 metri, acțiunile fiind sincronizate.
4.	Deplasare liniară aproximativ 3.5 metri a dronelor R1 și R2, pe axa X la înălțimea de 1 metru, acțiune sincronizată cu deplasarea liniară a celorlalte 2 drone R3 și R4 la înălțimea de 2 metri.
5.	Aterizare simultană în regiuni prestabilite.
6.	Deplasarea a 3 drone cu survolare de regiuni (1, 2, respectiv 4 regiuni), cu moment de sincronizare în ultima regiune vizitată (Fig. 17). Regiunile vizitate de drona R2 sunt poziționate la înălțimi diferite.
7.	Deplasare liniară urmată de un moment de sincronizare comun pentru toate cele 4 drone, urmat de survolarea unor regiuni (la înălțimi diferite față de înălțimea de decolare) și deplasarea către locurile de aterizare, cu traiectorii care se intersectează de câte două ori (Fig. 18).

8.	Scenariu ce include evitarea de obiecte (câte 1 obiect pentru R2 și R3, respectiv 2 obiecte pentru R1) și survolare de regiuni. Sunt descrise două momente de sincronizare, ce au ca scop prioritizarea deplasării pe segmentele de intersecție ale traiectoriilor (Fig. 19).

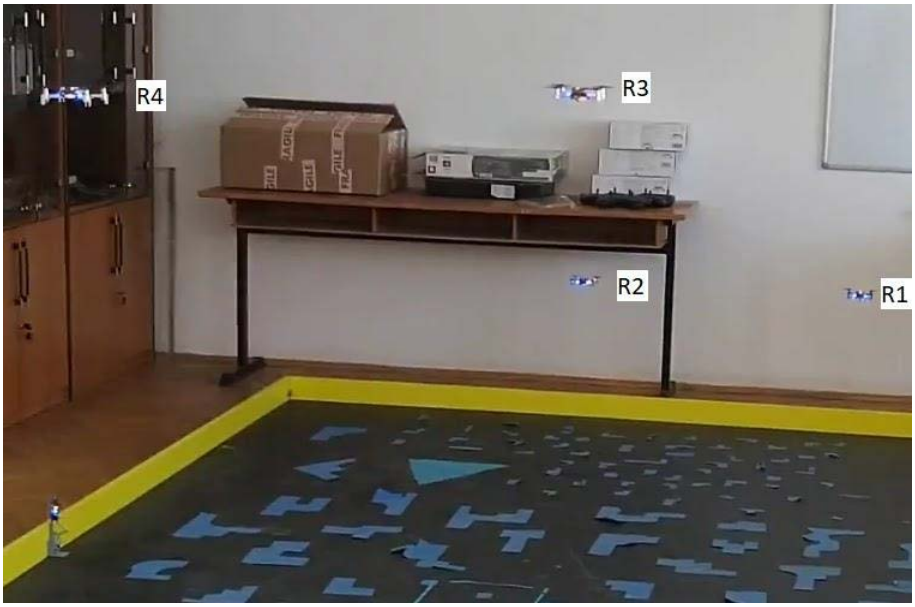

Fig. 15: Testare acțiuni sincronizate cu 2 echipe a câte 2 drone

Deplasarea între regiuni se realizează liniar, centrul regiunii ce trebuie survolată fiind la capătul unui segment care marchează deplasarea. Scenariile în care există mai multe obstacole de diferite forme implică parcurgerea unor traiectorii prestabilite de către 3 drone Crazyflie. Obstacolele au fost realizate din polistiren expandat de culoare neagră (Fig. 16).

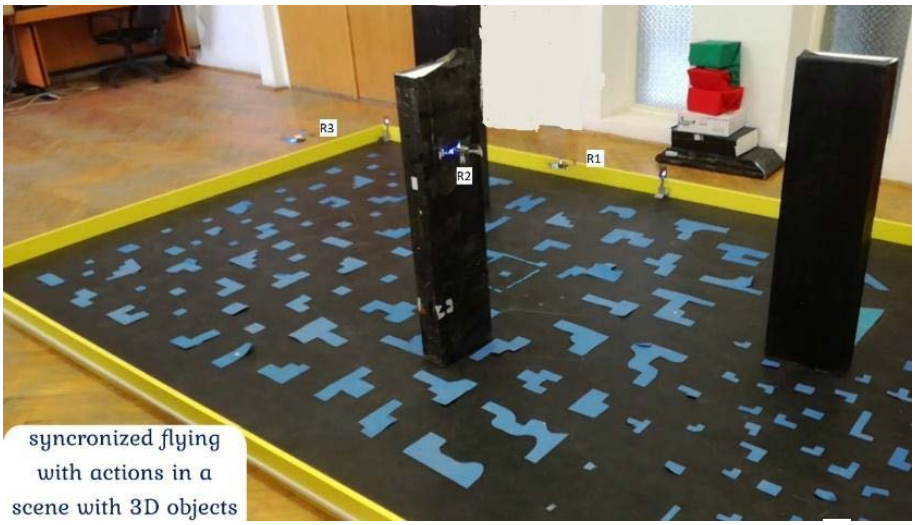

Fig. 16: Poziționarea obstacolelor în scenariul cu evitare de obstacole

Experimentele descrise mai sus pot fi vizualizate accesând link-ul: https://www.youtube.com/watch?v=tig0uLlNRXo.

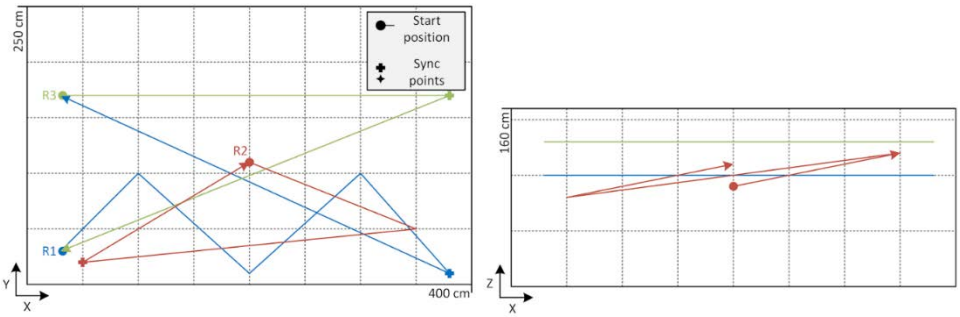

Fig. 17: Scenariu de test cu intersecții multiple și un singur moment de sincronizare

În Fig. 17 sunt reprezentate traiectoriile parcurse de drone în spațiul de lucru, fiind marcate punctele de plecare și punctele de sincronizare, conform descrierii din Tabelul 2. Traiectoria parcursă de fiecare dronă este descrisă prin segmente de drepte.

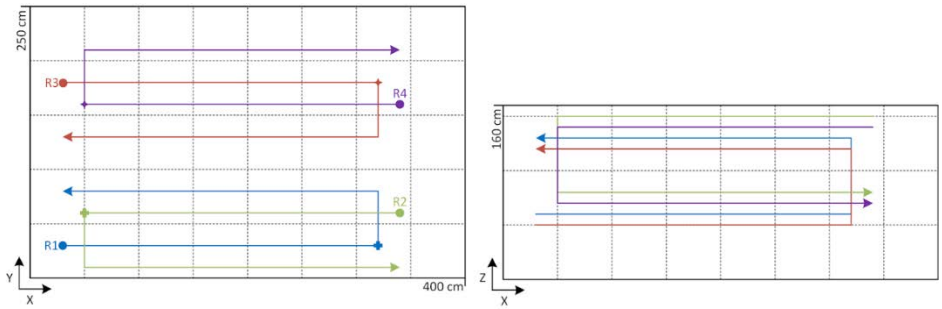

Fig. 18: Scenariu de test cu survolare de regiuni la înălțimi diferite

Au fost efectuate mai multe încercări cu poziționări diferite ale obstacolelor în spațiul de lucru, traiectoriile parcurse de cele 3 drone fiind reprezentate în Fig. 19, împreună cu două momente de sincronizare.

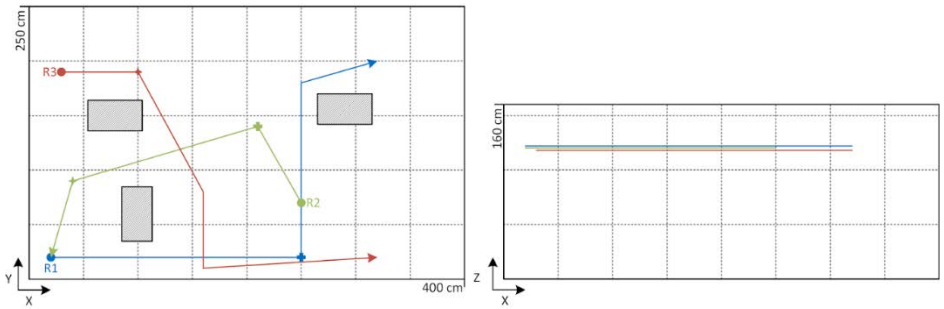

Fig. 19: Scenariu de test cu evitare de obstacole și două momente de sincronizare

Soluțiile de planificare propuse returnează secvențe de mișcare și de efectuare a acțiunilor de către roboții echipei, împreună cu momente necesare de sincronizare. Pentru experimentele bazate pe soluțiile de planificare propuse a fost aleasă o soluție de poziționare mixtă, bazată pe sistemele LPS și Flowdeck. Testele efectuate au relevat un comportament stabil al dronelor, cu ușoare oscilații doar în vecinătatea punctelor de destinație, respectiv cu o ușoară derivă în cazul zborului la punct fix. Integrarea algoritmilor de planificare într-un instrument software și validarea în timp real contribuie la creșterea impactului cercetărilor efectuate, permițând totodată dezvoltări ulterioare ce îmbină aspectele teoretice și experimentale.

Bibliografie

[1] Loco-positioning-system for Crazyflie 2/0, available at https://www.bitcraze.io/getting-started-with-the-loco-positioning-system/.

[2] E. Karapistoli F. N. Pavlidou I. Gragopoulos I. Tsetsinas "An overview of the IEEE 802.15.4a Standard" IEEE Communications Magazine vol. 48 no. 1 pp. 47-53 Jan. 2010.

[3] Measurements of accuracy and precision of the Loco Positioning system, available at https://wiki.bitcraze.io/misc:investigations:lps-precision.

[4] Cristian Mahulea, Marius Kloetzer, Ramón González, *Path Planning of Cooperative Mobile Robots Using Discrete Event Models*, Wiley-IEEE Press, IEEE Press Series on Systems Science and Engineering, series editor MengChu Zhou, ISBN 978-1-119-48632-9, 2020.

[5] Cristian Mahulea, Marius Kloetzer, Jean-Jacques Lesage, *Multi-robot Path Planning with Boolean Specifications and Collision Avoidance*, 15th Workshop on Discrete Event Systems (WODES'20), Rio de Janeiro, Brazil, 2020 (accepted).

[6] Cristian Mahulea, Marius Kloetzer, *Robot Planning Based on Boolean Specifications Using Petri Net Models*, IEEE Transactions on Automatic Control (TAC), vol. 63 (7), pp. 2218-2225, 2018.

[7] Marius Kloetzer, Cristian Mahulea, *Path Planning for Robotic Teams based on LTL Specifications and Petri Net Models*, Discrete Event Dynamic Systems, vol. 30 (1), pp. 55-79, 2020.

[8] Cristian Mahulea, Eduardo Montijano, Marius Kloetzer, *Distributed Multirobot Path Planning in Unknown Maps Using Petri Net Models*, 21st IFAC World Congress, Berlin, Germany, 2020 (accepted).

[9] M. Kloetzer, A. Burlacu, G. Enescu, S. Caraiman, C. Mahulea, Optimal Indoor Goods Delivery Using Drones, 24th IEEE International Conference on Emerging Technologies and Factory Automation (ETFA), pp. 1579-1582, 2019.

[10] C. Budaciu, N. Botezatu, M. Kloetzer, A. Burlacu: On the Evaluation of the Crazyflie Modular Quadcopter System, *IEEE 24th International Conference on Emerging Technologies and Factory Automation (ETFA)*, Zaragoza, Spain, 2019.