tatea "Politehnica" Timişoara
de Electronică şi Telecomunicaţii
mentul de Electronică Aplicată

Teza de doctorat

Contribuţii la conducerea adaptivă a roboţilor prin prelucrarea informaţiei vizuale utilizând reţele neuronale celulare

Conducător de doctorat

prof. dr. ing. Tiponuţ Virgil

Doctorand

ing. Gacsádi Alexandru

Timişoara, 2001

Mulțumiri

În primul rând mulțumesc conducătorului științific prof. dr. ing. Virgil Tiponuț care mi-a fost dascăl, de ajutorul căruia m-am bucurat la fiecare solicitare. Sub conducerea dânsului am avut posibilitatea de a studia în domenii de știință actuale și de a descoperi frumusețea activității de cercetare.

Doresc să mulțumesc domnului acad. dr. ing. Roska Tamás și domnului prof. dr. ing. Szolgay Péter pentru generozitatea lor, pentru ajutorul și resursele materiale valoroase puse la dispoziție în cadrul stagiilor efectuate la Laboratorul de Calcul Analogic și Neuronal de la Institutul de Calculatoare și Automatizări din Budapesta.

Aduc mulțumiri referenților științifici prof. dr. ing. Mircea Ivănescu și prof. dr. ing. Toma Leonida Dragomir pentru observațiile prețioase și sugestiile constructive prin care au contribuit la îmbunătățirea conținutului științific al tezei.

Mulțumesc domnului prof. dr. ing. Teodor Maghiar pentru înțelegerea și sprijinul de care am beneficiat în permanență la locul de muncă în cadrul Universității din Oradea.

Pentru încurajarea și ajutorul primit în pregătirea doctoratului se cuvin mulțumiri domnului prof. dr. Bondor Károly și domnului prof. dr. ing. Teodor Leuca.

Vreau de asemenea să mulțumesc Colectivului Departamentului de Electronică Aplicată din Timișoara pentru sprijinul acordat pe parcursul anilor.

Mulțumesc profesorilor, prietenilor și colegilor mei, care m-au ajutat în spațiu și în timp mai apropiat sau mai îndepărtat.

Nu în ultimul rând, mulțumesc familiei pentru înțelegerea, răbdarea și sprijinul prin care au participat la elaborarea acestei lucrări.

Oradea, 29.01.2001

Gacsádi Alexandru

Cuprins

1. Introducere

Rețeaua neuronală celulară (CNN-Cellular neural network [7,8]) este un procesor analogic cu structură bidimensională sau multidimensională, fiind formată din circuite analogice neliniare identice dispuse regulat, numite celule. Aceste celule interacționează local între ele pe baza unor operatori de care depinde rezultatul prelucrării. Adăugând la aceste procesoare elementare unități de memorie locală, unități de calcul aritmetic şi logic respectiv o unitate de programare globală se obține calculatorul universal CNN (CNN-Universal Machine [87]).

Prin implementarea în tehnologia VLSI a acestor calculatoare CNN, prelucrarea semnalului cu rețele neuronale celulare devine complet paralelă şi realizabilă în timp real. Astfel, din punctul de vedere al vitezei de calcul, procesarea CNN a semnalelor analogice poate constitui o alternativă avantajoasă față de procesarea semnalelor cu calculatoarele numerice seriale, sau față de alte mijloace hard specializate de procesare paralelă. În anumite clase de aplicații viteza de procesare a semnalelor cu calculatorul CNN poate fi cu până la trei ordine de mărime mai mare față de varianta de prelucrare numerică convențională.

Rețele neuronale celulare se pot utiliza cu succes într-o gamă foarte largă de aplicații [13], un domeniu semnificativ fiind modelarea şi analiza sistemelor biologice [102,119]. Prin modelarea funcționării retinei este posibilă înțelegerea, verificarea şi exprimarea unor noi ipoteze cu privire la funcționarea acesteia [90]. Pe de altă parte, cunoştințele dobândite permit elaborarea principiilor teoretice ce stau la baza funcționării retinei artificiale [111].

Puterea de calcul deosebită a rețelelor neuronale celulare, echivalentă în domeniul digital cu cca. 10^{12} operații pe secundă, se poate exemplifica şi prin capacitatea acestora de a rezolva fără iterații, printr-o singură instrucțiune elementară, ecuații diferențiale [62,91].

Având în vedere structura bidimensională, specifică rețelelor neuronale celulare, rezultă corespondența imediată a fiecărei celule cu un element discret de imagine. De aceea, rețele neuronale celulare se pot utiliza cu succes în prelucrări de imagini, mai ales la procedee care solicită o mare putere de calcul [16,17]. Lucrările publicate în acest domeniu demonstrează posibilitatea utilizării rețelelor neuronale celulare începând de la

prelucrări elementare ale imaginilor [1,71], segmentare [66,86], scheletizare [49], recunoaştere de forme [93], codarea imaginilor [63,98], criptografie [20], realizarea de memorii asociative [3,101], metode de prelucrare a imaginilor bazate pe morfologia matematică [120] şi până la prelucrări complexe spațiio-temporale [95].

În sens Turing calculatorul CNN este universal, adică prin utilizarea lui se poate rezolva în principiu orice clasă de probleme [11,12,18,109]. Soluționarea cu metode CNN a unei probleme concrete se poate realiza numai după ce aceasta a fost transpusă în mediul CNN. Pentru anumite aplicații complexe nu este posibilă determinarea unui singur set de operatori care să asigure prelucrarea dorită cu ajutorul rețelei neuronale celulară într-o singură etapă şi să ofere un rezultat convenabil. În astfel de cazuri, un set de operatori nu este altceva decât o singură instrucțiune elementară iar proiectarea CNN se transformă de fapt în programare CNN cu ajutorul a mai multor instrucțiuni elementare sau subrutine având la bază operatori existenți [20,63,66,92,93,98,100,109].

Practic în orice aplicație care se poate prezenta sub formă de algoritm este posibilă şi utilizarea rețelelor neuronale celulare. Însă este evident că, deşi procesarea CNN este aplicabilă în orice problemă, utilizarea acesteia nu este eficientă, față de prelucrarea numerică serială, decât într-o clasă restrânsă de probleme. În anumite situații este posibil ca vitezele de prelucrare a celor două metode să fie comparabile. Dacă problema de soluționat reprezintă o parte a unui algoritm complex atunci alegerea variantei de procesare cu rețele neuronale celulare respectiv prelucrarea numerică serială, se face pe baza performanțelor în ansamblu ale celor două procedee preferându-se o structură de prelucrare cât mai omogenă. Metoda de prelucrare CNN se va alege numai dacă majoritatea subrutinelor din algoritm pot fi prelucrate eficient cu rețele neuronale celulare. Astfel, devine important ca în cadrul unor algoritmi complecşi fiecare subrutină să se poată realiza cu metode CNN, chiar dacă aparent sunt cu performanțe asemănătoare față de prelucrarea numerică serială.

Cercetările actuale în domeniul rețelelor neuronale celulare sunt orientate pe două direcții principale, interdependente, una cu caracter teoretic şi alta cu caracter aplicativ.

Pe plan teoretic, un obiectiv important de cercetare este tocmai dezvoltarea unor algoritmi analogici CNN competitivi. Aplicații utilizând rețele neuronale celulare se regăsesc în cele mai diverse domenii din ştiință, pentru soluționarea acelor probleme care se pot reduce la prelucrări de semnale bidimensionale. În unele aplicații, procesarea CNN s-a dovedit a fi chiar superioară privind viteza de lucru, față de alte metode care utilizează procesarea numerică serială.

Posibilitatea realizării unor circuite integrate CNN, chiar şi pe baza tehnologiei VLSI actuale, se datorează proprietății specifice rețelei neuronale celulare şi anume existența numai a unor interacțiuni locale între celule. În acest fel s-a putut evidenția practic avantajul major al procesării CNN complet paralele, care rezultă în cazul implementării pe chip. An de an, în numeroase laboratoare de cercetare din întreaga lume se proiectează, se realizează şi se testează circuite integrate specializate care implementează rețele neuronale celulare. Aceste circuite prezintă parametri din ce în ce mai performanți, făcând astfel mai atractiv sub aspect aplicativ mediul de lucru CNN [21,57,68,78].

Pentru o problemă concretă, dintr-o oarecare soluție posibilă, procedeul CNN poate deveni chiar soluția potrivită, cea mai eficientă, numai după implementarea algoritmilor CNN şi funcționarea lor robustă pe chip.

Fără posibilitatea implementării pe chip a operatorilor şi algoritmilor CNN, acestea nu ar fi decât simple programe de simulare scrise pentru un microprocesor inexistent. Importanța algoritmilor CNN creşte considerabil numai după ce sunt validați în urma implementării acestora pe circuite CNN reale.

În cazul implementării unor algoritmi simpli CNN, chiar şi circuitul CNN cu numai 20*22 de celule s-a dovedit mai eficient din punct de vedere al vitezei de procesare față de prelucrarea serială [21]. La proiectarea de noi circuite CNN se intenționează nu numai creşterea dimensiunilor rețelei dar şi extinderea performanțelor acestora, prin utilizarea de imagini mască sau de imagini de polarizare respectiv creşterea preciziei şi a robusteții în funcționare [68].

În domeniul cercetărilor aplicative privind rețelele neuronale celulare, prezintă interes deosebit nu numai proiectarea şi realizarea de circuite CNN dar şi dezvoltarea de mijloace hard şi soft utile în elaborarea de aplicații cu asemenea rețele neuronale. Astfel, se menționează mediul de dezvoltare CNN "CadetWin" (CNN application development environment and toolkit under Windows [123]), limbajul propriu de nivel superior "Alpha" (CNN Alpha language and compiler [129]) şi limbajul de asamblare "AMC" (Extended analogic macro code and interpreter language and compiler [130]). Aceste mijloace sunt deosebit de utile începând de la elaborările din faza de cercetare teoretică până la evaluarea rezultatelor experimentale pe circuite CNN.

Trebuie subliniată strânsa interdependență între dezvoltarea aspectelor teoretice din domeniul CNN şi realizările aplicative susținute de existența unor circuite CNN

performante. Cele două direcții de cercetare se află nu numai într-o competiție continuă, dar şi într-o susținere reciprocă.

Rețelele neuronale în general, fiind mijloace adaptive de procesare paralele şi distribuită, sunt utilizate cu succes în multe domenii cum ar fi: modelarea adaptivă, recunoaşterea de forme, recunoaşterea vorbirii sau a scrisului, predicția, clasificarea, optimizarea, comanda roboților şi alte aplicații în care problemele sunt imprecise prin însăşi natura lor sau descrierea prin metode analitice este dificilă [22,51,52,104,105].

Comanda roboților pe baza informației vizuale sau comanda vizuală ("visual servoing") [15,50,54,80,110,112]) este un domeniu multidisciplinar care include prelucrarea imaginilor, cinematica şi dinamica roboților, teoria sistemelor, fiecare dintre acestea fiind domenii de cercetare de sine stătătoare. Pentru a se putea realiza în timp real comanda vizuală a roboților, este necesară o putere de calcul deosebit de ridicată, accesibilă totuşi datorită evoluției rapide a calculatoarelor seriale ori prin crearea unor arhitecturi paralele specializate de procesare.

Ca o altă alternativă față de tehnicile convenționale, rețelele neuronale pot fi utilizate la rezolvarea problemelor specifice de la comanda adaptivă a robotului [67,73,82], dar şi la controlul pe baza informației vizuale [4,26,107]. S-a arătat că prin utilizarea rețelelor neuronale se poate realiza o poziționare vizuală cu un grad de acuratețe similar cu cel obținut prin tehnici analitice. Rețelele neuronale constituie de asemenea, o disciplină de sine stătătoare, dar prin introducerea acesteia în robotică se intenționează realizarea în timp real a comenzii, creşterea adaptibilității şi a preciziei de poziționare a robotului şi nicidecum apariția unei dificultăți în plus în abordarea unui domeniu de ştiință suficient de complex.

Prelucrarea rapidă şi robustă a imaginii este deosebit de importantă pentru creşterea adaptabilității şi obținerea unei viteze ridicate la comanda vizuală a robotului, de aceea rămâne un subiect de cercetare de certă actualitate [54].

Alături de celelalte aplicații prezentate anterior, rețelele neuronale celulare s-au dovedit eficiente şi în alte probleme specifice de prelucrare a imaginilor care se pot utiliza la comanda vizuală a robotului, începând de la extracția unui obiect din imagine [114], detecția de poziție, mişcare, viteză [63,66,69,89,113], navigarea robotului într-un mediu cu obstacole [56,94,98], ori la conducerea propriu-zisă [60].

Scopul cercetării prezentate în această lucrare îl constituie studiul posibilității implementării rețelelor neuronale celulare la conducerea adaptivă a unui robot pe baza informației vizuale.

În cadrul studiului acestor domenii de cercetare se au în vedere atât aspectele legate de creşterea capacităţii de procesare a semnalelor bidimensionale cu reţele neuronale celulare cât şi aspectele legate de posibilităţile dezvoltării comenzii adaptive a roboţilor bazată pe imagini.

În capitolul 2 se prezintă structura de bază a reţelei neuronale celulare, dar şi alte structuri importante derivate din aceasta, care s-au dovedit utile în cele mai diverse aplicaţii. În acest capitol se face o prezentare generală a noţiunilor teoretice legate de reţelele neuronale celulare. Sunt prezentate definiţii, notaţii şi expresii utilizate apoi în capitolele următoare. În finalul acestui capitol se prezintă conceptul general de maşină universală CNN, respectiv cele două variante de chip-CNN care s-au utilizat pentru validarea experimentală a noilor metode de procesare propuse de autor în cadrul acestei teze.

Începând cu capitolul 3 sunt prezentate rezultatele unor cercetări proprii, originale ale autorului în legătură cu posibile utilizări ale reţelelor neuronale celulare la prelucrarea de semnale bidimensionale cu aplicaţii în robotică. Rezultatele obţinute în aceste domenii oferă răspunsuri la unele întrebări cu caracter teoretic şi aplicativ.

La elaborarea noilor algoritmi CNN s-a ţinut cont permanent de posibilitatea implementării acestora cu uşurinţă pe cel mai performant chip CNN existent la ora respectivă, având în vedere adevărul demult cunoscut şi anume că, performanţele reale practice sunt cele mai edificatoare pentru verificarea oricărei teorii.

În capitolul 3 este prezentat un algoritm CNN original, bazat pe imagini, de urmărire a unui obiect în mişcare cu ajutorul unei camere video montate pe braţul unui robot cu două grade de libertate. Avantajul major al controlului bazat pe imagini faţă de controlul bazat pe poziţii constă în precizia poziţionării sistemului care este mai puţin sensibil la erorile de calibrare ale camerei video. Soluţiile propuse pentru fiecare dintre etapele algoritmului pot fi utilizate cu succes şi în alte aplicaţii similare, chiar dacă metodele utilizate pentru prelucrarea imaginilor depind întotdeauna de aplicaţia concretă, având în vedere că, pentru comanda vizuală a robotului informaţiile apriorice sunt deosebit de importante.

Astfel, în prima etapă a algoritmului este prezentată selectarea şi urmărirea într-o imagine a unui singur model special cu o suprafaţă şi viteză redusă, caz în care se poate utiliza metoda CNN bazată pe suprafaţă, metodă suficient de robustă pentru un set mare de distorsiuni ale imaginii şi mişcări reduse. Un pas important al algoritmului este determinarea punctului central al unui obiect dintr-o imagine, pentru care se poate

utiliza metoda cunoscută deja din literatura de specialitate, dar pentru soluționarea acestei probleme, se prezintă şi un alt nou procedeu original CNN mai rapid. Pentru conducerea brațului de robot bazată pe imagini s-au introdus imaginile (semnalele bidimensionale) de comandă, structura acestora fiind definită pentru mai multe clase de aplicații şi tipuri de circuite CNN.

Determinarea imaginilor de comandă necesare pentru conducerea vizuală bazată pe imagini a unui robot se poate realiza şi prin interpolare. De aceea, în capitolul 4 se studiază şi se propun metode noi, originale, de interpolare a semnalelor bidimensionale, folosind rețele neuronale celulare. Operatorii utilizați la interpolare s-au obținut pe cale analitică şi sunt numai operatori liniari de reacție, de dimensiune 3*3, fiind posibilă astfel implementarea imediată a acestui procedeu pe un chip-CNN existent [68]. Astfel, prelucrarea semnalelor se poate efectua în timp real, prin reducerea a numărului de încărcări pe chip a imaginilor inițiale. În acest capitol sunt prezentate, de asemenea, rezultatele verificării experimentale a acestor metode de interpolare la creşterea rezoluției imaginilor binare şi cu niveluri de gri, la codarea şi decodarea imaginilor în vederea transmiterii lor şi la reconstrucția de imagini.

În capitolul 5 se propune o metodă originală CNN de planificare a traiectoriei unui robot mobil cu reacție vizuală bazată pe imagini, într-un mediu de lucru cu obstacole cunoscute, statice sau mobile. La prelucrarea imaginilor, pentru planificarea traiectoriei robotului se utilizează numai rețele neuronale celulare, asigurând obținerea în timp real a mărimii de comandă necesară robotului. Navigarea robotului ocolind obstacolele, de la o poziție inițială de start până la o poziție țintă pe o traiectorie optimă ca lungime şi cu număr minim de viraje, se realizează prin metoda globală CNN. Metoda globală CNN, asociată cu nivelul ierarhic superior din conducerea robotului, poate oferi traiectoria planificată optimă nivelului ierarhic inferior din conducerea robotului la navigarea prin metodă locală.

Concluziile finale, contribuțiile autorului şi unele direcții de cercetare viitoare sunt prezentate în capitolul 6. Programele scrise în limbaj de asamblare AMC, utilizate pentru soluționarea problemelor descrise în lucrare sunt prezentate în anexele 1-6.

2. Rețeaua neuronală celulară

În acest capitol se face o prezentare generală a principalelor noțiuni legate de rețelele neuronale celulare, începând cu structura de bază (CNN-Cellular neural network [7,8]), introdusă prima oară de L.O. Chua şi L. Yang în 1988. De atunci, din punct de vedere teoretic şi aplicativ acest domeniu s-a îmbogățit continuu şi cu alte variante de rețelele neuronale celulare, care au dovedit o eficiență deosebită în procesarea spațiio-temporală a semnalelor bidimensionale, în timp real. Pe parcursul capitolului se fac referiri la noțiunile teoretice privind asigurarea stabilității rețelei şi la metodele actuale utilizate pentru determinarea operatorilor necesari la prelucrarea semnalelor cu rețele neuronale celulare. În centrul atenției aplicațiilor cu rețele neuronale celulare stă posibilitatea implementării algoritmilor pe calculatorul universal CNN (CNN-Universal Machine [87]), prin care prelucrarea devine complet paralelă şi realizabilă în timp real. Sunt prezentate, de asemenea, două variante de circuite integrate CNN experimentale, împreună cu mediul de dezvoltare de aplicații "CadetWin" (CNN application development environment and toolkit under Windows [123]), care s-au utilizat la simularea şi la testarea pe chip a unor noi algoritmi propuşi în capitolele următoare.

2.1 Rețeaua neuronală celulară de bază

Definiția 1: Structura rețelei neuronale celulare de bază

Rețeaua neuronală celulară de bază [7,8,14] este o structură bidimensională rectangulară, fiind formată din circuite analogice identice neliniare, dispuse regulat, numite celule.

În figura 2.1 se prezintă structura unei rețele neuronale celulară de bază monostrat cu M linii şi N coloane, de dimensiune M*N. Se notează cu (i,j) coordonatele spațiale carteziene ale celulei C(i,j) din rețeaua neuronală celulară, $1 \le i \le M; 1 \le j \le N$.

Definiția 2: Vecinătatea sau sfera de influență a unei celule

Prin vecinătatea sau sfera de influență de rangul r a unei celule C(i,j) se înțelege mulțimea:

$$N_r(i,j) = \{C(k,l) \mid \max\{|k-i|,|l-j|\} \le r\}, \qquad (2.1)$$

unde r este un număr întreg pozitiv.

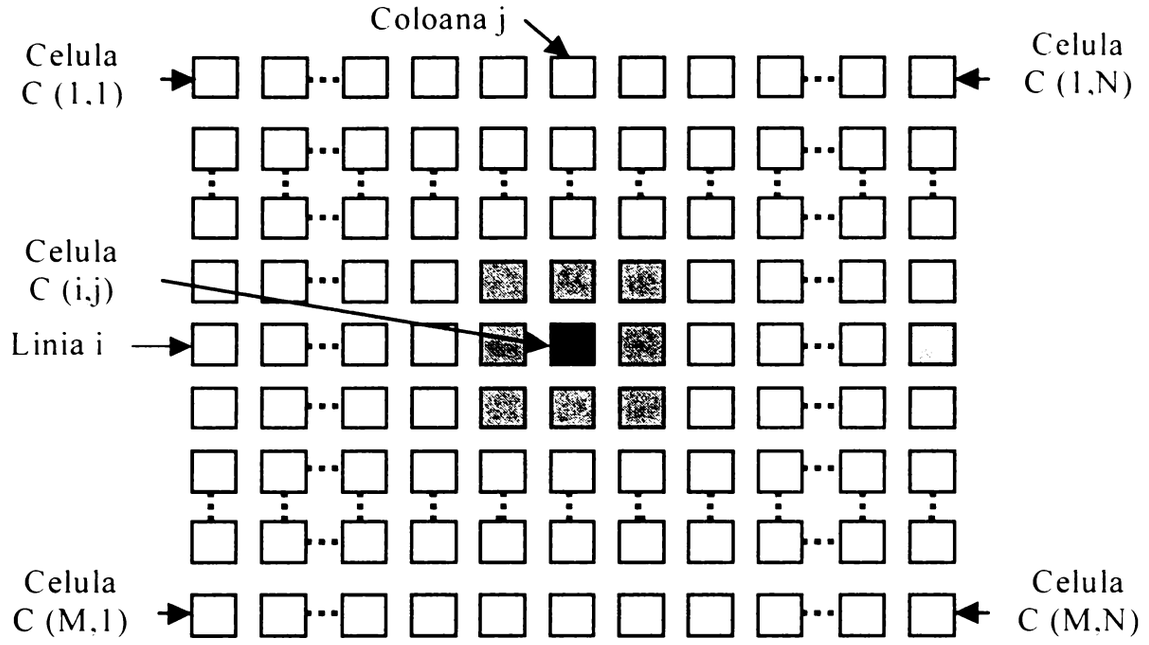


Figura 2.1. Structura de bază a unei rețele neuronale celulare monostrat cu M linii şi N coloane.

Pentru o celulă C(i,j) dimensiunea sferei de influență N_r de rază r este (2r+1)*(2r+1). În figura 2.2 este prezentată sfera de influență a unei celule C(i,j) de rază r=1 (vecinătate 3*3), respectiv de rază r=2 (vecinătate 5*5).

Sistemul de vecinătăți prezintă proprietatea de simetrie, dacă presupunând că, $C(i,j) \in N_r(k,l)$ atunci şi $C(k,l) \in N_r(i,j)$, pentru oricare C(i,j) şi C(k,l) din rețeaua neuronală celulară.

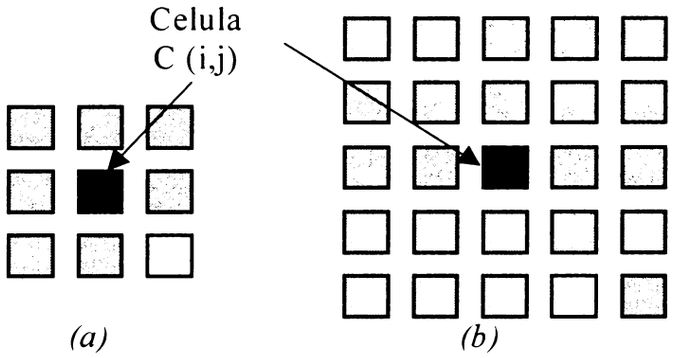


*Figura 2.2. Vecinătatea unei celule C(i,j): a) de rază r=1 (vecinătate 3*3); b) de rază r=2 (vecinătate 5*5).*

O celulă este conectată în mod direct numai cu acele celule din rețea care se află în sfera ei de influență. Conexiunile dintre celule se realizează pe baza unor operatori denumiți "template" în literatura de specialitate CNN. Aceşti operatori au semnificația unor matrice de pondere sau de conexiune.

Observații:

1.) Celulele neconectate în mod direct între ele pot comunica datorită efectului de propagare a interacțiunilor locale dintre celule, pe durata regimului tranzitoriu din rețea. Astfel, cu toate că interconectările dintre celule sunt numai locale, se constată că utilizând rețelele CNN pot fi extrase și proprietăți globale ale semnalelor procesate.

2.) Pentru r=N-1 și M=N se obține rețeaua neuronală celulară complet conectată, unde fiecare celulă este conectată cu fiecare celulă, adică sfera de influență pentru o celulă este toată rețeaua. În această situație rezultă de fapt rețeaua neuronală de tip Hopfield [51].

Definiția 3: Celule interne, celule de margine și extreme

Se consideră C(i,j) o celulă internă a unei rețele dacă toate celulele C(k,l) $\in N_r(i,j)$ din vecinătatea acesteia sunt incluse în acea rețea. În caz contrar, celulele sunt de margine sau extreme.

În figura 2.3 se prezintă pozițiile celulelor extreme și de margine.

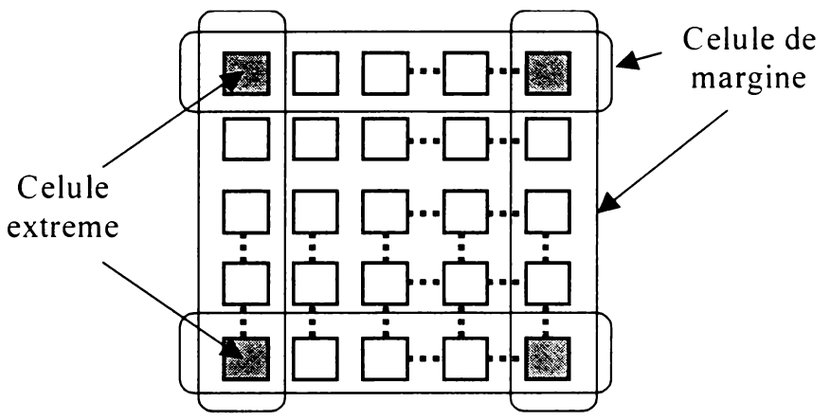


Figura 2.3. Pozițiile celulelor extreme și de margine.

Definiția 4: Circuitul electric de bază al unei celule interne

Structura circuitului electric aferent fiecărei celule C(i,j) este prezentată în figura 2.4.

Se constată prezența a 3 noduri caracterizate respectiv de tensiunea de intrare v_{uij}, tensiunea de stare v_{xij} și tensiunea de ieșire v_{yij}.

Circuitul electric al fiecărei celule C(i,j) include următoarele elemente:

- sursa de tensiune independentă E_{ij}, care constituie mărimea de intrare a celulei;
- o sursă de curent continuu independentă I, reprezentând polarizarea;
- sursele liniare de curent comandate în tensiune, I_{xu}, I_{xy}, și I_{yx};

- rezistențele R_x, R_y și o capacitate C. Parametri elementelor de circuit I, C, R_x, R_y sunt identici pentru fiecare celulă.

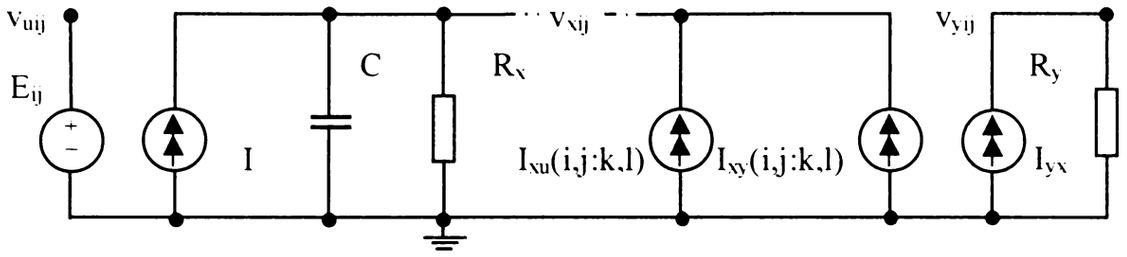


Figura 2.4. Structura circuitului electric pentru o celulă C(i,j).

Sursele de curent I_{xu}, I_{xy} sunt comandate cu tensiunile de intrare v_{ukl} (reacție directă) respectiv cu tensiunile de ieşire v_{ykl} (reacția inversă) ale celulelor $C(k,l) \in N_r(i,j)$ din vecinătatea de rază r a celulei C(i,j.

Valoarea curentului furnizat de sursele comandate este dată de relațiile:

$$I_{xu}(i,j;k,l)=B(i,j;k,l)v_{ukl}\,, \tag{2.2}$$

$$I_{xy}(i,j;k,l)=A(i,j;k,l)v_{ykl}\,, \tag{2.3}$$

în care $C(i,j) \in N_r(k,l)$ pentru $1 \le i \le M; 1 \le j \le N$, iar A şi B sunt operatori sau template. A(i,j;k,l) se numeşte operatorul de comandă al ieşirii prin reacție inversă denumit pe scurt operator de reacție, iar B(i,j;k,l) este operator de comandă al intrării prin reacție directă, sau operator de comandă.

Sursa neliniară de curent comandată în tensiune, de la ieşire, este caracterizată de relația:

$$I_{yx} = \frac{1}{R_y} f(v_{xij})\,, \tag{2.4}$$

$$\text{în care } f(v) = \frac{1}{2}\left[|v+1| - |v-1|\right]. \tag{2.5}$$

Reprezentarea grafică a funcției f(v) este prezentată în figura 2.5.

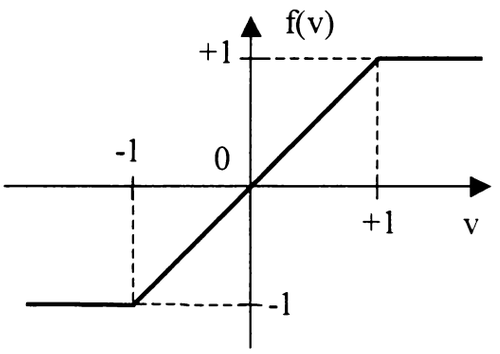


Figura 2.5. Caracteristica de transfer f(v).

Definiția 5: Rețeaua neuronală celulară standard

Evoluția stării fiecărei celule interne C(i,j) este caracterizată de următoarele relații, care rezultă pe baza figurii 2.4:

- ecuația de stare (2.6)

$$C\frac{dv_{xij}(t)}{dt} = -\frac{1}{R_x}v_{xij}(t) + \sum_{C(k,l)\in N_r(i,j)} A(i,j;k,l)v_{ykl}(t) + \sum_{C(k,l)\in N_r(i,j)} B(i,j;k,l)v_{ukl}(t) + I$$

- ecuația de intrare

$$v_{uij} = E_{ij}; \quad (2.7)$$

- ecuația de ieşire

$$v_{yij}(t) = \frac{1}{2}\left[\left|v_{xij}(t)+1\right| - \left|v_{xij}(t)-1\right|\right]; \quad (2.8)$$

- condiții inițiale (2.9)

$$\left|v_{xij}(0)\right| \le 1 \text{ şi } \left|v_{uij}(t)\right| \le 1;$$

$$A(i,j;k,l) = A(k,l;i,j);$$

$$C > 0 \text{ şi } R_x > 0,$$

pentru $1 \le i,k \le M$ şi $1 \le j,l \le N$ cu $C(k,l) \in N_r(i,j)$.

Observații:

1.) Alegând valori convenabile, din considerente tehnologice, pentru elementele de circuit R_x, R_y şi C, există posibilitatea implementării VLSI a unor astfel de rețele neuronale celulare.

2.) Valoarea maximă a tensiunii de stare pentru orice celulă şi în orice moment pe durata regimului tranzitoriu este limitată [7]:

$$\left|v_{ij}(t)\right|_{max} = V_{max} = 1 + R_x I + R_x \max\left[\sum_{C(k,l)\in N_r(i,j)} \left\{\left|A(i,j;k,l)\right| + \left|B(i,j;k,l)\right|\right\}\right]. \quad (2.10)$$

3.) Structura bidimensională a unei rețele neuronale de bază sugerează ideea asocierii acesteia cu o imagine, fiecare celulă corespunzând unui pixel. Mai mult decât atât, pot fi asociate imagini corespunzătoare stării rețelei (STATE), intrării (INPUT) şi respectiv ieşirii (OUTPUT). În această lucrare se va utiliza pentru modelul unei rețele neuronale celulare această asociere cu imagini.

4.) Pentru respectarea condițiilor inițiale impuse unei rețele neuronale celulare, relațiile (2.9), valorile elementelor de imagine se normează după cum urmează:

- în cazul unei imagini binare se asociază pentru nivelul alb valoarea –1 şi valoarea +1 pentru nivelul negru al unui pixel;
- la o imagine cu niveluri de gri (gray-scale) se asociază pentru nivelul pixelilor de la alb la negru domeniul de valori [-1,+1] (domeniul valorilor standard CNN).

5.) Circuitul electric are constanta de timp a regimului tranzitoriu: $\tau_{CNN}=C*R_x$.

2.2 Rețea neuronală celulară invariantă în spațiu

În expresiile matematice (2.2)-(2.9), care descriu evoluția în timp a stării rețelei neuronale celulare, s-au folosit notațiile v pentru potențialul într-un nod de circuit şi I în cazul surselor de curent, pentru a sublinia posibilitatea reală a implementării acestora pe un chip [13,76]. În scopul simplificării descrierii rețelelor neuronale celulare se utilizează frecvent forma normalizată, în care R_x, R_y şi C au valori unitare.

În acest caz evoluția stării fiecărei celule interne C_{ij}, pentru $1 \le i \le M; 1 \le j \le N$, va fi caracterizată de următoarele expresii:

- ecuația de stare

$$\dot{x} = \frac{dx_{ij}}{dt} = -x_{ij} + \sum_{C_{kl} \in N_r} A_{ij,kl} y_{kl} + \sum_{C_{kl} \in N_r} B_{ij,kl} u_{kl} + z_{ij}; \qquad (2.11)$$

- ecuația de ieşire

$$y_{ij} = f(x_{ij}) = \frac{1}{2}\left[\left|x_{ij} + 1\right| - \left|x_{ij} - 1\right|\right]; \qquad (2.12)$$

- condiții inițiale

$$\left|x_{ij}(0)\right| \le 1 \text{ şi } \left|u_{ij}(t)\right| \le 1, \qquad (2.13)$$

unde u_{ij} reprezintă intrarea, x_{ij} reprezintă starea, iar y_{ij} reprezintă ieşirea.

Operatorul de reacție s-a notat cu $A_{ij.kl}$, $B_{ij.kl}$ reprezintă operatorul de comandă, iar z_{ij} semnifică polarizarea, ("threshold", "bias" sau "current").

Celulele C_{kl} sunt din vecinătatea de rază r a celulei C_{ij}, adică $C_{kl} \in N_r$.

Definiția 6: Rețea neuronală celulară invariantă în spațiu

O rețea neuronală celulară este invariantă în spațiu sau izotropică dacă operatorii $A_{ij.kl}$, $B_{ij.kl}$, z_{ij}, nu depind de indicii (i,j).

În acest caz:

$$\sum_{C_{kl} \in N_r} A_{ij,kl} y_{kl} = \sum_{|k-i| \le r} \sum_{|l-j| \le r} A_{k-i,l-j} y_{kl}, \qquad (2.14)$$

$$\sum_{C_{kl} \in N_r} B_{ij,kl} u_{kl} = \sum_{|k-i| \le r} \sum_{|l-j| \le r} B_{k-i,l-j} u_{kl}, \qquad (2.15)$$

$$z_{ij} = z. \qquad (2.16)$$

Mărimile $A_{ij.kl}$, $B_{ij.kl}$ şi z_{ij} se vor considera invariante în spațiu şi în timp în toate cazurile, dacă nu se specifică altfel.

Operatorii frecvent utilizați în rețelele neuronale celulare pot fi clasificați după cum urmează [14]:

1.) Operatorul liniar de reacție $A_{ij.kl}$ cu elemente constante din mulțimea numerelor reale, $a_{kl} \in \boldsymbol{R}$.

2.) Operatorul liniar de comandă $B_{ij.kl}$ cu elemente constante din mulțimea numerelor reale, $b_{kl} \in \boldsymbol{R}$.

3.) Operatorul neliniar de reacție $A_{ij.kl}$ pentru care:

$A_{ij.kl}y_{kl}=a(y_{ij})$, funcție de valoarea la ieşire a celulelor, y_{ij},

$A_{ij.kl}y_{kl}=a(y_{kl})$, funcție de valoarea la ieşire a celulelor, y_{kl},

$A_{ij.kl}y_{kl}=a(y_{ij} \oplus y_{ij})$, funcție de combinațiile valorilor de ieşire a celulelor, y_{ij}, y_{kl}, unde "$\oplus$" poate reprezenta operația de adunare, scădere sau produs.

4.) Operatorul neliniar de comandă $B_{ij.kl}$ pentru care:

$B_{ij.kl}u_{kl}=b(u_{ij})$, funcție de valoarea la intrarea celulelor, u_{ij},

$B_{ij.kl}u_{kl}=b(u_{kl})$, funcție de valoarea la intrarea celulelor, u_{kl},

$B_{ij.kl}u_{kl}=b(u_{ij} \oplus u_{ij})$, funcție de combinațiile valorilor de intrare a celulelor, u_{ij}, u_{kl}, unde $\oplus$ poate reprezenta operația de adunare, scădere, sau produs.

5.) Se poate introduce operatorul de tip $C_{ij.kl}$ care este de forma:

$C_{ij.kl}x_{kl}= c(x_{ij})$, funcție de valoarea de stare a celulelor, x_{ij},

$C_{ij.kl}x_{kl}= c(x_{kl})$, funcție de valoarea de stare a celulelor, x_{kl},

$C_{ij.kl}x_{kl}= c(x_{ij} \oplus x_{ij})$, funcție de combinațiile valorilor de stare a celulelor, x_{ij}, x_{kl}, unde "$\oplus$" poate reprezenta operația de adunare, scădere, sau produs.

6.) Uneori se utilizează operatorul cel mai general, de tip $D_{ij.kl}$, care este de forma:

$D_{ij.kl}=d(u_{ij}, x_{ij}, y_{ij}, u_{kl}, x_{kl}, y_{kl})$.

În argumentul funcției d, între valorile celulelor de intrare, de stare şi de ieşire poate exista operația de adunare, scădere, sau produs.

Având în vedere tipurile de operatori prezentați mai sus, ecuația diferențială de stare pentru rețeaua neuronală celulară standard devine:

$$\dot{x} = -x_{ij} + \sum_{C_{kl} \in N_r} A_{ij,kl}y_{kl} + \sum_{C_{kl} \in N_r} B_{ij,kl}u_{kl} + z_{ij} + \sum_{C_{kl} \in N_r} C_{ij,kl}x_{kl} + \sum_{C_{kl} \in N_r} D_{ij,kl}(u_{kl}, x_{kl}, y_{kl}). \quad (2.17)$$

2.3 Rețele neuronale celulare multistrat

Noțiunile definite în cazul rețelelor bidimensionale pot fi generalizate cu uşurință pentru cazul cu mai multe dimensiuni, rezultând astfel rețelele multidimensionale [7.8.48].

Definiția 7: Rețele neuronale celulare multistrat

Pentru o rețea neuronală celulară multistrat, ecuația de stare poate fi exprimată sub forma vectorială:

$$\boldsymbol{C}\frac{\mathrm{d}\,x_{ij}(t)}{\mathrm{d}t} = -\boldsymbol{R}_x^{-1} \otimes \boldsymbol{x}_{ij}(t) + \boldsymbol{A} \otimes \boldsymbol{y}_{ij}(t) + \boldsymbol{B} \otimes \boldsymbol{u}_{ij} + z_{ij}\,, \tag{2.18}$$

în care "⊗" reprezintă operatorul de convoluție spațială. Numărul L al straturilor, este un întreg pozitiv, $L \in \boldsymbol{N}$, iar indicele stratului este l, l= 1, 2,…,L.

Capacitatea $\boldsymbol{C}$, şi rezistențele $\boldsymbol{R_x}$ sunt matrice diagonale de forma:

$$\boldsymbol{C} = \begin{bmatrix} C_1 & 0 & 0 & 0 & 0 \\ 0 & C_2 & 0 & 0 & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ 0 & 0 & C_l & 0 & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ 0 & 0 & 0 & C_{L-1} & 0 \\ 0 & 0 & 0 & 0 & C_L \end{bmatrix}, \tag{2.19}$$

$$\boldsymbol{R_x} = \begin{bmatrix} R_{x1} & 0 & 0 & 0 & 0 \\ 0 & R_{x2} & 0 & 0 & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ 0 & 0 & R_{xl} & 0 & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ 0 & 0 & 0 & R_{xL-1} & 0 \\ 0 & 0 & 0 & 0 & R_{xL} \end{bmatrix}, \tag{2.20}$$

iar operatorii $\boldsymbol{A}, \boldsymbol{B}$ sunt matrice triunghiulare:

$$\boldsymbol{A} = \begin{bmatrix} A_{1,1} & A_{1,2} & A_{1,1} & A_{1,L-1} & A_{1,L} \\ A_{2,1} & A_{2,2} & A_{2,1} & A_{2,L-1} & A_{2,L} \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ A_{1,1} & A_{1,2} & A_{1,1} & A_{1,L-1} & A_{1,L} \\ \vdots & & & \vdots & \\ A_{L-1,1} & A_{L-1,2} & A_{L-1,2} & A_{L-1,L-1} & A_{L-1,L} \\ A_{L,1} & A_{L,2} & A_{L,1} & A_{L,L-1} & A_{L,L} \end{bmatrix}, \tag{2.21}$$

$$\boldsymbol{B} = \begin{bmatrix} B_{1,1} & B_{1,2} & B_{1,1} & B_{1,L-1} & B_{1,L} \\ B_{2,1} & B_{2,2} & B_{2,1} & B_{2,L-1} & B_{2,L} \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ B_{1,1} & B_{1,2} & B_{1,1} & B_{1,L-1} & B_{1,L} \\ \vdots & \vdots & & \vdots & \\ B_{L-1,1} & B_{L-1,2} & B_{L-1,2} & B_{L-1,L-1} & B_{L-1,L} \\ B_{L,1} & B_{L,2} & B_{L,1} & B_{L,L-1} & B_{L,L} \end{bmatrix}. \tag{2.22}$$

Mărimile $\boldsymbol{u_{ij}}$, $\boldsymbol{x_{ij}}$, $\boldsymbol{y_{ij}}$ $\boldsymbol{z_{ij}}$ sunt date de expresiile:

$$\boldsymbol{u_{ij}} = \begin{bmatrix} u_{ij1} \\ u_{ij2} \\ \vdots \\ u_{ijl} \\ \vdots \\ u_{ijL-1} \\ u_{ijL} \end{bmatrix}, \boldsymbol{x_{ij}} = \begin{bmatrix} x_{ij1} \\ x_{ij2} \\ \vdots \\ x_{ijl} \\ \vdots \\ x_{ijL-1} \\ x_{ijL} \end{bmatrix}, \boldsymbol{y_{ij}} = \begin{bmatrix} y_{ij1} \\ y_{ij2} \\ \vdots \\ y_{ijl} \\ \vdots \\ y_{ijL-1} \\ y_{ijL} \end{bmatrix}, \boldsymbol{z_{ij}} = \begin{bmatrix} z_{ij1} \\ z_{ij2} \\ \vdots \\ z_{ijl} \\ \vdots \\ z_{ijL-1} \\ z_{ijL} \end{bmatrix}. \tag{2.23}$$

Observații:

1.) În majoritatea cazurilor, pentru elementele R_x, R_y și C din diferitele straturi, se utilizează valori identice.

2.) Caracteristicile rețelelor multistrat, cum ar fi stabilitatea, sunt similare cu cele ale rețelelor monostrat.

3.) Dacă nu există operatori de reacție între straturi, funcția realizată de rețeaua neuronală multistrat se poate implementa prin prelucrare secvențială cu o rețea neuronală monostrat.

2.4 Alte variante de rețele neuronale celulare

Pentru unele aplicații mai complexe pot fi definite variante speciale de rețele neuronale celulare.

1.) În cazul cel mai general, mărimile $A_{ij,kl}$, $B_{ij,kl}$, z_{ij} sunt variabile în spațiu (depind de coordonatele (i,j)) respectiv se pot modifica în funcție de variabila de timp [48].

2.) Dacă mărimea de polarizare z_{ij} se face dependentă de poziția din rețea, rezultă o posibilitate suplimentară pentru a comanda rețeaua neuronală celulară. Imaginea asociată ("bias-map") influențează regimul tranzitoriu al rețelei [59] şi prin urmare determină, alături de imaginile de intrare respectiv de stare, comportamentul rețelei.

3.) Operatorul de reacție $A_{ij,kl}$ este cu întârziere [88]:

$A^{\tau}_{ij,kl} y_{kl}(t-\tau)$ cu posibilitatea dependenței întârzierii de poziție, adică: $\tau=\tau_{kl}$.

4.) Operatorul de comandă $B_{ij,kl}$ este cu întârziere [88]:

$B^{\tau}_{ij,kl} y_{kl}(t-\tau)$ cu posibilitatea dependenței întârzierii de poziție, adică: $\tau=\tau_{kl}$.

5.) Rețele neuronale celulare cu imagine mască ("MASK image").

În cazul unei rețele neuronale celulare de dimensiune M*N, se asociază imaginii de stare, o imaginea mască, având aceiaşi dimensiune. Pentru anumite valori ale pixelilor din imaginea mască nu se permite modificarea în timpul procesului de prelucrare a valorilor elementelor de imagine din pozițiile identice ale imaginii de stare.

De exemplu, considerând cazul unei imagini mască binară, pixelii din imaginea de stare îşi pot modifica valoarea în timpul procesului tranzitoriu, dacă în imaginea mască le corespunde în aceiaşi poziție un element cu valoare +1. Dacă în imaginea mască un pixel este de valoare –1, elementul identic ca poziție din imaginea de stare nu-şi va putea schimba valoarea.

6.) Prin discretizare temporală, ecuația diferențială de stare a unei celule poate fi aproximată cu o ecuație cu diferențe finite: (2.24)

- $$\frac{1}{h}\left[x_{ij}((n+1)h)-x_{ij}(nh)\right]=-x_{ij}(nh)+\sum_{C_{kl}\in N_r} A_{ij,kl}y_{kl}(nh)+\sum_{C_{kl}\in N_r} B_{ij,kl}u_{kl}(nh)+z_{ij}$$

În relația de mai sus t=n*h este variabila discretă de timp, h este pasul de eşantionare iar n este un număr natural. Totodată funcția de transfer f(v), relația (2.5), se modifică în consecință, astfel că mărimea la ieşire va fi dată de relația: (2.25)

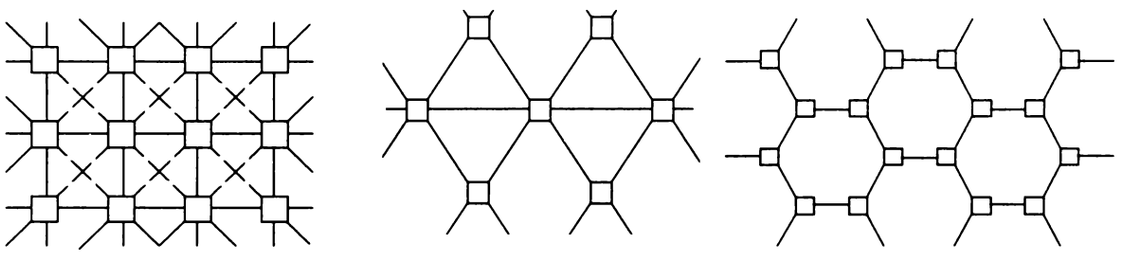
- $y_{ij}(nh) = 0.5(|x_{ij}(nh)+1| - |x_{ij}(nh)-1|) \equiv f[x_{ij}(nh)]$, pentru $1 \le i \le M; 1 \le j \le N$.

Pe baza acestor ecuații, operația realizată de rețea poate fi interpretată ca fiind similară unui filtru neliniar bidimensional, care transformă o imagine de stare x(nh) într-o altă imagine x[(n+1)h]. Dacă operatorii (A, B, z) sunt invarianți în spațiu filtrul este de asemenea invariant în spațiu. Deoarece filtrarea realizată între două momente discrete succesive se poate folosi doar la detecția de proprietăți locale, pentru extragerea de proprietăți globale sunt necesare mai multe iterații.

Vecinătatea N_{nr} este de n ori mai mare decât vecinătatea N_r astfel încât pentru un număr n de iterații suficient de mare, vecinătatea N_{nr}, va cuprinde toată imaginea indiferent de valorile lui (i,j) şi r. Aşadar, proprietatea de propagare oferă posibilitatea extragerii şi a unor caracteristici globale ale imaginilor, cu păstrarea proprietăților locale, vecinii apropiați având efecte multiple față de cei îndepărtați.

7.) Pentru funcțiile de transfer de la stare către ieşire, pot fi folosite în afară de funcția f(v) liniară pe porțiuni şi alte funcții cum ar fi: funcția sigmoidă, funcția Gauss funcția signum, funcția cu pantă negativă.

8.) Dispunerea celulelor în strat respectiv modul de interconectare a unei celule cu celulele din vecinătate poate căpăta şi alte forme decât cea specifică rețelei cu structură rectangulară. Unele exemple sunt prezentate în figura 2.6 [88].



(a) *(b)* *(c)*

Figura 2.6. Structuri de vecinătăți CNN: (a) structura rectangulară; (b) structura triunghiulară; (a) structura hexagonală.

9.) Interacțiunile dintre celule pot fi perturbate uneori de zgomote. Pentru a evidenția acest lucru, se poate însuma în paralel cu intrarea, cu starea sau cu ieşirea fiecărei celule o sursă de curent de zgomot, $\eta_{ij}(t)$.

2.5 Rețeaua neuronală celulară discretă în timp – DTCNN

Definiția 8: Rețeaua neuronală celulară discretă în timp

Rețeaua neuronală celulară discretă în timp (DTCNN-Discrete time cellular neural network) [47,48] este definită de relațiile:

- ecuația de stare

$$v_x^c = R_x(A_d^c v_y^d(kT) + B_d^c v_u^d + I^c); \tag{2.26}$$

- ecuația de ieşire

$$v_y^c(kT) = f(v_x^c((k-1)T))) = \begin{cases} +1 & \text{dacă} \quad v_x^c(k-1) > 0 \\ -1 & \text{dacă} \quad v_x^c(k-1) < 0 \end{cases}. \tag{2.27}$$

În expresiile de mai sus v_u^c, v_x^c, v_y^c reprezintă tensiunile din nodurile circuitului electric corespunzător unei celule "c" din rețea, A_d^c, B_d^c, I^c reprezintă operatorul de reacție, de comandă, respectiv polarizarea corespunzătoare unei celule "c" din rețea, cu vecinătatea "d".

În figura 2.7 se prezintă schema electrică corespunzătoare unei celule din rețeaua neuronală celulară discretă în timp. Prin (kT) a fost notată variabila de timp discretă.

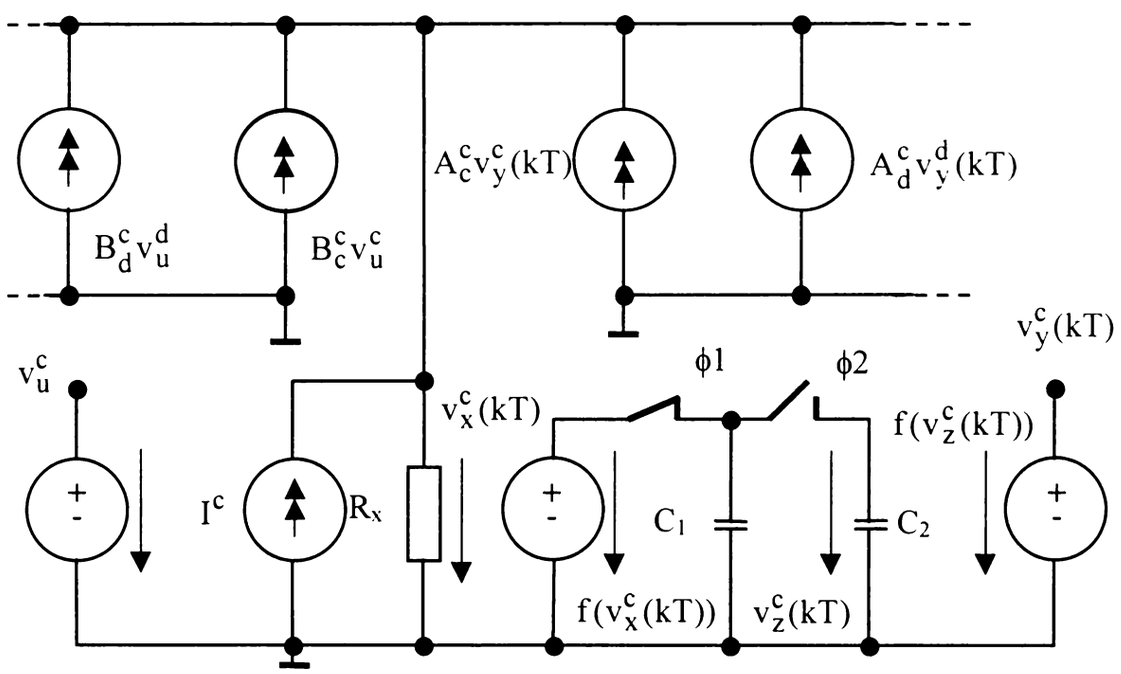


Figura 2.7. Schema electrică corespunzătoare unei celule din rețeaua neuronală celulară discretă în timp – DTCNN.

La rețeaua neuronală celulară discretă în timp pentru fiecare secvență de timp se obține o stare stabilă de ieşire, fapt ce simplifică considerabil simularea pe calculator a comportării rețelei [29]. O iterație completă se obține la finele perioadei semnalelor de comandă Φ1 şi Φ2 (figura 2.8), după cum urmează.

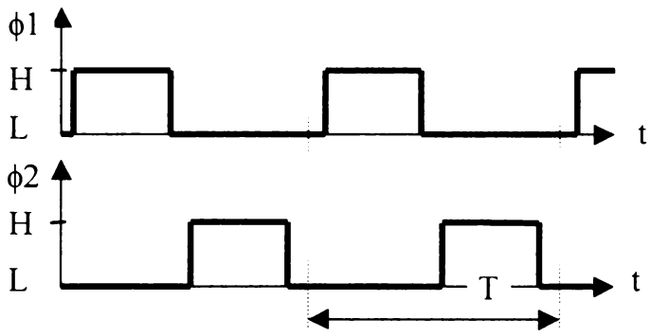


Figura 2.8. Semnalele de tact pentru o rețea discretă în timp.

Dacă Φ1 este pe nivel "H" capacitatea C_1 se încarcă la nivelul $+V_{sat}$ sau $-V_{sat}$. Când Φ2 atinge nivelul "H" capacitățile C_1 şi C_2 vor fi conectate în paralel şi în consecință:

$$v_z^c(kT) = \frac{C_1 f(v_x^c((k-1)T)) + C_2 f(v_z^c((k-1)T))}{C_1 + C_2}. \qquad (2.28)$$

Sursele de tensiune controlate în tensiune $f(v_x^c(kT))$ şi $f(v_z^c(kT))$ au caracteristici neliniare definite de relația:

$$f(v) = \begin{cases} V_{sat} & \text{dacă} \quad v > 0 \\ -V_{sat} & \text{dacă} \quad v < 0 \end{cases} \qquad (2.29)$$

Observații:

1.) Față de rețeaua neuronală celulară analogică de bază, în cazul rețelei neuronale celulare discrete în timp ieşirile pot avea numai valorile binare +1 sau –1.

2.) Înaintea unei iterații complete se poate defini şi imaginea de ieşire $y^c(0) \in [-1, 1]$. Aceasta se poate considera ca o nouă imagine inițială, alături de imaginea de intrare şi imaginea de stare.

3.) Operatorii pot fi continui, dependenți sau independenți de variabila timp.

4.) Rețelele neuronale celulare discrete în timp sunt robuste la toleranțele elementelor de circuit.

5.) Prin modificarea frecvenței semnalelor de tact se poate controla viteza de lucru a rețelei.

2.6 Rețeaua neuronală celulară standard invariantă în spațiu cu operatori de dimensiune 3*3

În marea majoritate a aplicațiilor se utilizează rețeaua neuronală celulară standard invariantă în spațiu şi cu operator de dimensiune 3*3, sau operatori reductibile la această dimensiune. În aceste cazuri se folosesc următoarele convenții de notare [13,14]:

- vecinătatea de rază 1 a unei celule C_{ij}, adică $C_{kl} \in N_1$,

$$\boldsymbol{C}_{kl} = \begin{bmatrix} c_{i-1,j-1} & c_{i-1,j} & c_{i-1,j+1} \\ c_{i,j-1} & c_{i,j} & c_{i,j+1} \\ c_{i+1,j-1} & c_{i+1,j} & c_{i+1,j+1} \end{bmatrix}; \tag{2.30}$$

- operatorul de reacție:

$$\boldsymbol{A} = \begin{bmatrix} a_{-1,-1} & a_{-1,0} & a_{-1,+1} \\ a_{0,-1} & a_{0,0} & a_{0,+1} \\ a_{+1,-1} & a_{+1,0} & a_{+1,+1} \end{bmatrix}; \tag{2.31}$$

- operatorul de comandă

$$\boldsymbol{B} = \begin{bmatrix} b_{-1,-1} & b_{-1,0} & b_{-1,+1} \\ b_{0,-1} & b_{0,0} & b_{0,+1} \\ b_{+1,-1} & b_{+1,0} & b_{+1,+1} \end{bmatrix}, \tag{2.32}$$

$$\begin{aligned} \sum_{C_{kl} \in N_r} A_{ij,kl} y_{kl} &= \sum_{|k-i| \le 1} \sum_{|l-j| \le 1} A_{k-i,l-j} y_{kl} = \sum_{k=-1}^{1} \sum_{l=-1}^{1} a_{kl} y_{i+k,j+l} \\ &= a_{-1,-1} y_{i-1,j-1} + a_{-1,+1} y_{i-1,j+1} + a_{+1,+1} y_{i+1,j+1} + a_{+1,-1} y_{i+1,j-1} + \\ &+ a_{-1,0} y_{i-1,j} + a_{+1,0} y_{i+1,j} + a_{0,-1} y_{i,j-1} + a_{0,-1} y_{i,j+1} + a_{0,0} y_{i,j} \overset{\text{def}}{=} \boldsymbol{A} \otimes \boldsymbol{Y}_{ij} \end{aligned} \tag{2.33}$$

$$\begin{aligned} \sum_{C_{kl} \in N_r} B_{ij,kl} u_{kl} &= \sum_{|k-i| \le 1} \sum_{|l-j| \le 1} B_{k-i,l-j} u_{kl} = \sum_{k=-1}^{1} \sum_{l=-1}^{1} b_{kl} u_{i+k,j+l} \\ &= b_{-1,-1} u_{i-1,j-1} + b_{-1,+1} u_{i-1,j+1} + b_{+1,+1} u_{i+1,j+1} + b_{+1,-1} u_{i+1,j-1} + \\ &+ b_{-1,0} u_{i-1,j} + b_{+1,0} u_{i+1,j} + b_{0,-1} u_{i,j-1} + b_{0,-1} u_{i,j+1} + b_{0,0} u_{i,j} \overset{\text{def}}{=} \boldsymbol{B} \otimes \boldsymbol{U}_{ij} \end{aligned} \tag{2.34}$$

În cele de mai sus s-a notat cu "⊗" operația de convoluție spațială.

Prin înlocuirea sumelor de corelație în expresia ecuației diferențiale de stare, corespunzătoare unei celule, rezultă:

$$\dot{x} = -x_{ij} + \boldsymbol{A} \otimes \boldsymbol{Y_{ij}} + \boldsymbol{B} \otimes \boldsymbol{U_{ij}} + z\,. \qquad (2.35)$$

Operația pe care o realizează o rețea neuronală celulară asupra unei imagini de intrare $\boldsymbol{U}(t_0)$ pentru obținerea unei imagini de ieşire stabilă $\boldsymbol{Y}$, este complet definită de operatorii $\boldsymbol{A}$, $\boldsymbol{B}$, z, precizați la rândul lor prin cei 19 parametri. Aceste elemente alcătuiesc o instrucțiune elementară CNN.

Pentru o rețea neuronală celulară standard, invariantă în spațiu cu operatori de dimensiune 3*3, în funcție de natura celor trei operatori se pot defini următoarele structuri:

1.) Rețeaua neuronală celulară standard de tip $C(\boldsymbol{A}\neq\boldsymbol{0}, \boldsymbol{B}\neq\boldsymbol{0}, z\neq 0)$, este caracterizată de prezența legăturilor sinaptice inhibitorii sau excitatorii având în vedre că:

$$\boldsymbol{A}\neq\boldsymbol{0},\ \boldsymbol{B}\neq\boldsymbol{0},\ z\neq 0. \qquad (2.36)$$

În figura 2.9 se prezintă circulația informației într-o rețea neuronală celulară standard, invariantă în spațiu cu operatori de dimensiune 3*3, iar structura unei celule C_{ij} este prezentată în figura 2.10.

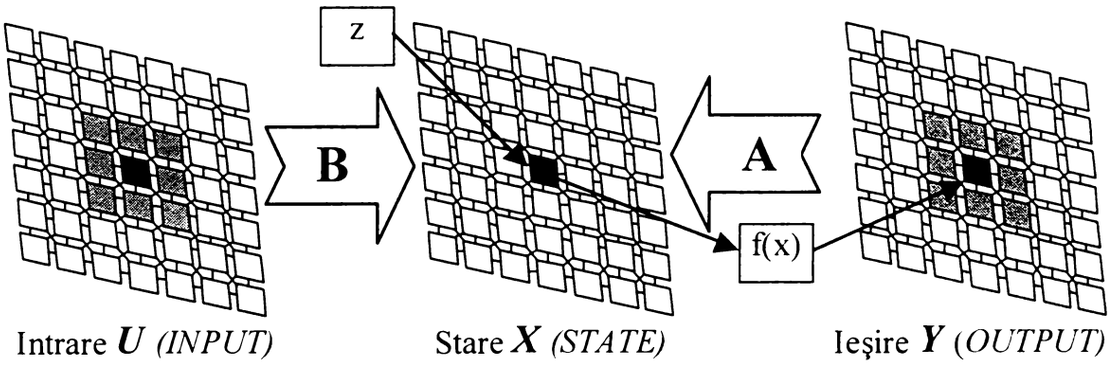


*Figura 2.9. Circulația informației într-o rețea neuronală celulară standard, invariantă în spațiu cu operatori de dimensiune 3*3.*

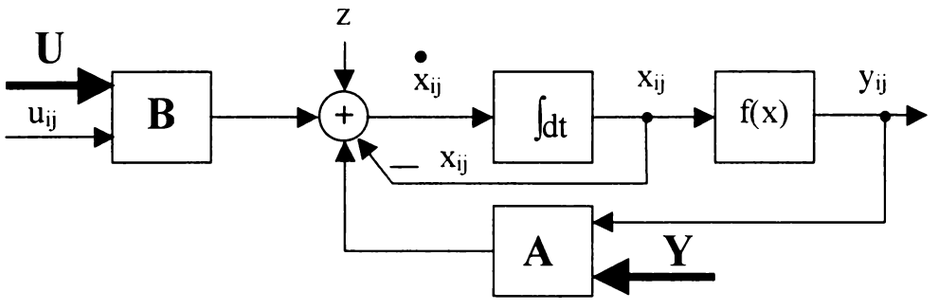


Figura 2.10. Structura unei celule C_{ij}.

2.) Rețea neuronală celulară de tip $C(\boldsymbol{0}, \boldsymbol{B\neq 0}, z\neq 0)$, numită cu cuplaj înainte sau fără reacție inversă și care este caracterizată de relațiile:

$$\boldsymbol{A=0},\ \boldsymbol{B\neq 0},\ z\neq 0. \tag{2.37}$$

3.) Rețea neuronală celulară de tip $C(\boldsymbol{A\neq 0}, \boldsymbol{B=0}, z\neq 0)$, numită cu cuplaj înapoi sau fără intrare și care este caracterizată de relațiile:

$$\boldsymbol{A\neq 0},\ \boldsymbol{B=0},\ z\neq 0. \tag{2.38}$$

4.) Rețea neuronală celulară de tip scalar, $C(\boldsymbol{A}=a_{00}, \boldsymbol{B\neq 0}, z\neq 0)$, este caracterizată de relațiile:

$$\boldsymbol{A}=a_{00},\ \boldsymbol{B\neq 0},\ z\neq 0. \tag{2.39}$$

Observație:

Un operator complet (template) include operatorul A de reacție inversă, operatorul B de comandă și polarizarea z. În cadrul acestei lucrări, pentru un operator complet se va utiliza convenția de notare: <nume operator>.tem [125].

2.7 Condiţiile iniţiale şi de frontieră ale reţelei neuronale celulare

Pentru o reţea neuronală celulară, ecuaţiile de stare ale celulelor formează un sistem de ecuaţii diferenţiale. Starea finală la ieşirea la reţelei, rezultă prin rezolvarea acestui sistem. Rezolvarea ecuaţiilor diferenţiale ordinare este posibilă numai dacă sunt cunoscute condiţiile iniţiale şi de limită sau de frontieră. Condiţiile de frontieră se pot realiza prin conectarea unor celule virtuale la marginile reţelei, atât pentru imaginea de stare cât şi pentru imaginea de ieşire [13,14,62].

În modelul unei reţele neuronale celulare starea iniţială a reţelei este asociată cu imaginea de stare la momentul iniţial t_0, adică cu $x_{ij}(t_0)$.

Definiţia 9: Celule virtuale

Celulele C_{kl} pentru care $|k-i \le r|, |l-j \le r|$, dar $k \notin \{1,2,....,M\}$ şi/sau $l \notin \{1,2,....,N\}$ se numesc celule virtuale. Acestor celule li se asociază intrări virtuale u_{kl}, stări virtuale x_{kl}, ieşiri virtuale y_{kl} şi polarizări virtuale z_{kl}, pentru ca sistemul de ecuaţii diferenţiale să devină compatibil.

În cazul unei reţele neuronale celulare de dimensiune M*N, includerea de celule virtuale presupune adăugarea a două linii şi a două coloane virtuale. Cele două linii virtuale, una sus şi una jos, vor avea indicele asociat pentru linii de 0 respectiv M+1. Cele două coloane virtuale, una în stânga şi una în dreapta reţelei, vor avea indicele asociat pentru coloane de 0 respectiv N+1. Pentru a impune anumite condiţii de frontieră, celulele virtuale vor fi forţate să aibă anumite valori, fapt ce se realizează prin conectarea lor la surse de tensiune. Sunt utilizate frecvent următoarele condiţii de frontieră [13]:

1.) Condiţii de frontieră cu valori constante (Dirichlet). (2.40)

- Celule virtuale din stânga: $y_{i0}=\alpha_1$ şi $u_{i0}=\beta_1$ pentru i =1,2,…,M;
- Celule virtuale din dreapta: $y_{iN+1}=\alpha_2$ şi $u_{iN+1}=\beta_2$ pentru i =1,2,…,M;
- Celule virtuale de sus: $y_{0j}=\alpha_3$ şi $u_{0j}=\beta_3$ pentru j =1,2,…,N;
- Celule virtuale de jos: $y_{M+1j}=\alpha_4$ şi $u_{M+1j}=\beta_4$ pentru j =1,2,…,N.

Mărimile α_1 , β_1, α_2 , β_2, α_3 , β_3, α_{14} , β_4, sunt în general constante identice având valorile E=0, E= –1, sau E= +1, aşa cum se arată în figura 2.11.

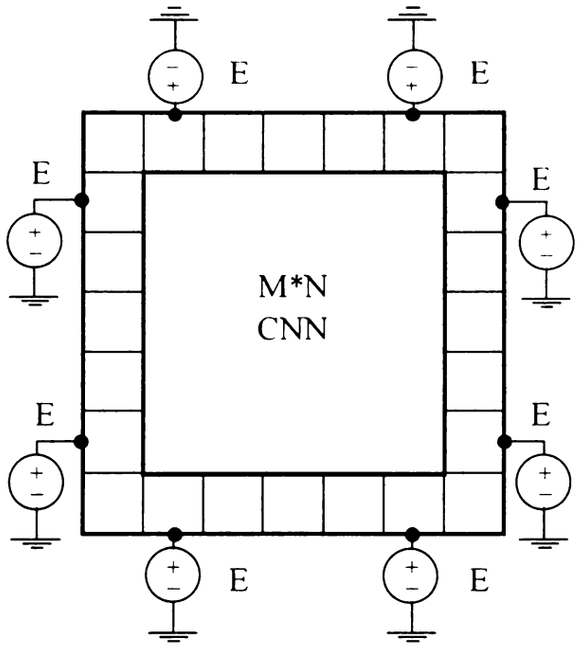


Figura 2.11. Condiții de frontieră cu valori constante.

2.) Condiția de frontieră zero-flux (Neumman). (2.41)

- Celule virtuale din stânga: $y_{i0}=y_{i1}$ şi $u_{i0}= u_{i1}$ pentru i =1,2,…,M;
- Celule virtuale din dreapta: $y_{iN+1}=y_{iN}$ şi $u_{iN+1}= u_{iN}$ pentru i =1,2,…,M;
- Celule virtuale de sus: $y_{0j}=y_{1j}$ şi $u_{0j}= u_{1j}$ pentru j =1,2,…,N;
- Celule virtuale de jos: $y_{M+1j}=y_{Mj}$ şi $u_{M+1j}= u_{Mj}$ pentru j =1,2,…,N.

Această condiție de frontieră este utilizată pentru cazul în care nu există intrare, existând doar stare inițială (figura 2.12).

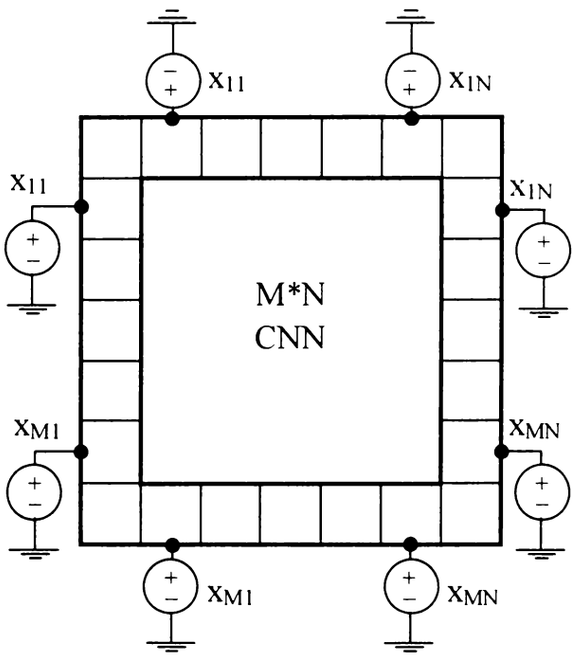


Figura 2.12. Condiția de frontieră zero-flux.

3.) Condiția de frontieră periodică (Toroidal). (2.42)

- Celule virtuale din stânga: $y_{i0}=y_{iN}$ şi $u_{i0}= u_{iN}$ pentru i =1,2,…,M;
- Celule virtuale din dreapta: $y_{iN+1}=y_{i1}$ şi $u_{iN+1}= u_{i1}$ pentru i =1,2,…,M;
- Celule virtuale de sus: $y_{0j}=y_{Mj}$ şi $u_{0j}= u_{Mj}$ pentru j =1,2,…,N;
- Celule virtuale de jos: $y_{M+1j}=y_{1j}$ şi $u_{M+1j}= u_{1j}$ pentru j =1,2,…,N.

În figura 2.13 se prezintă interpretarea condiției de frontieră periodică.

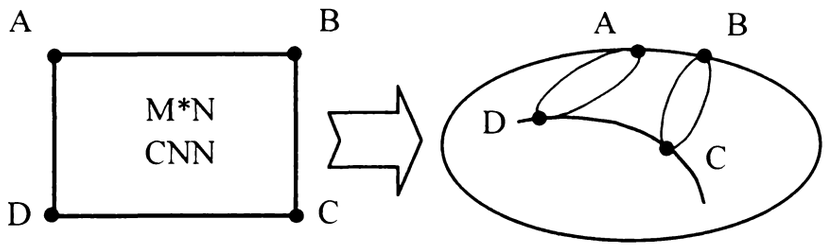


Figura 2.13. Condiția de frontieră periodică.

2.8 Stabilitatea rețelelor neuronale celulare

Dacă se aplică semnal la intrarea unei rețele neuronale celulare şi condițiile inițiale sunt bine determinate, este de dorit ca după un regim tranzitoriu cât mai scurt să se ajungă la o stare de echilibru staționar, adică starea celulelor să conveargă către o stare stabilă.

Un punct de echilibru stabil pentru o rețea neuronală celulară este definit ca vectorul de stare de echilibru care are componentele formate din stări stabile de echilibru ale celulelor.

Punctele de echilibru stabil sunt puncte ale unui bazin de atracție, mulțimea traiectoriilor dintre starea inițială şi starea finală staționară a sistemului fiind convergente către acest punct. Astfel, pentru o rețea neuronală celulară se pot poziționa un set de bazine de atracție în spațiul stărilor, în centrul bazinelor aflându-se punctele de echilibru stabil.

Definiția 10: Rețea neuronală celulară complet stabilă sau convergentă

O rețea neuronală celulară standard cu dimensiunea M*N, de tip $C(A,B,z)$ este complet stabilă sau convergentă dacă toate soluțiile $x(t,x_0)$ cu starea inițială x_0 converg către un punct de echilibru x_{ij}^*[7]. Se consideră x_{ij}^* stare stabilă de echilibru pentru starea unei celule x_{ij} din rețea dacă:

$$\left.\frac{dx_{ij}}{dt}\right|_{x_{ij}=x_{ij}^*} = 0, \tag{2.43}$$

$$\lim_{t\to\infty} x_{ij}(t) = x_{ij}^* \quad \text{pentru } 1 \le i \le M; \quad 1 \le j \le N. \tag{2.44}$$

Punctele de echilibru stabil depind în general de condițiile inițiale, de intrare şi de dinamica circuitului.

Comportarea dinamică depinde de interacțiunile spațiale locale. În cazul general, convergența rețelelor neuronale celulare către o stare stabilă este dependentă numai de operatorul de reacție inversă A, fiind independentă de operatorul de reacție directă B [7,9,10,76]. De aceea, studiul stabilității rețelelor neuronale celulare se reduce la studiul operatorilor A, structura acestora determinând dacă evoluția stării rețelei se va face sau nu către o stare stabilă.

O rețea neuronală celulară de dimensiune M*N, invariantă în spațiu şi având dimensiune arbitrară pentru vecinătăți este complet stabilă pentru o intrare şi polarizare cu valori constante, dacă sunt satisfăcute următoarele trei condiții [14]:

- Operatorul A este simetric în raport cu elementul central

  $A_{ij.kl}=A_{kl.ij}$. (2.45)

- Funcția de transfer neliniară de la stare la ieşire este derivabilă, mărginită şi

  $\dot{f}(x_{i,j})>0$ pentru $-\infty < x_{i,j} < +\infty$. (2.46)

- Toate punctele de echilibru sunt izolate. Un punct de echilibru x_{ij}^* pentru $\dot{x} = f(x)$ se spune că este izolat dacă şi numai dacă nu există un alt punct de echilibru într-o vecinătate suficient de mică a lui x_{ij}^*.

Dacă sunt îndeplinite aceste condiții înseamnă că toate valorile ieşirilor celulelor rețelei converg către o constantă, adică rețeaua este convergentă către o stare stabilă:

$$\lim_{t\to\infty} y_{ij}(t) = \text{const.} \quad \text{, pentru } 1 \le i \le M; \quad 1 \le j \le N. \tag{2.47}$$

Din punct de vedere practic este important ca alături de satisfacerea condiției de stabilitate să fie satisfăcută şi condiția $A_{ij.kl} > 1/R_x$. În această situație starea celulelor tind asimptotic către o valoare stabilă, care în modul este mai mare decât 1.

Dacă $\lim_{t\to\infty} |x_{ij}| > 1$ adică $x_{ij}^* > 1$ şi $\lim_{t\to\infty} y_{ij}(t) = \text{const.}$

rezultă $\lim_{t\to\infty} y_{ij}(t) = \pm 1$ pentru $1 \le i \le M; \quad 1 \le j \le N$. (2.48)

Aceste expresii au o semnificație importantă, deoarece presupunând că intrarea şi/sau starea inițială a rețelei sunt imagini binare, la sfârşitul procesului tranzitoriu se va obține la ieşire tot o imagine binară.

Dacă s-a determinat un operator de tip A invariant în spațiu şi liniar, pentru aprecierea satisfacerii criteriului de stabilitate nu este necesară cercetarea îndeplinirii condițiilor la fiecare operator în parte. Pentru aprecierea uşoară a satisfacerii de către operatori a criteriului de stabilitate, sunt definite clase de operatori după criteriul simetriei de semn a elementelor precum şi transformări ale acestora prin a căror utilizare se poate determina dacă rețeaua converge către o stare stabilă [7,9,10].

Dacă utilizând aceste transformări nu se poate demonstra sau verifica direct că operatorul asigură sau nu stabilitatea, aceasta nu înseamnă inevitabil că rețeaua este

instabilă, ci doar că metoda nu oferă un răspuns univoc privind satisfacerea criteriului de stabilitate.

Operatorii utilizați în cadrul prezentei teze de doctorat satisfac condițiile necesare pentru ca rețelele neuronale celulare implementate pe baza lor să prezinte un comportament stabil.

2.9 Proiectarea aplicațiilor cu rețele neuronale celulare

Proiectarea în domeniul rețelelor neuronale celulare înseamnă în primul rând proiectarea de operatori. Ca orice rețea neuronală şi rețeaua celulară are capacitatea de a învăța. Funcția dorită a se realiza prin prelucrarea cu rețele neuronale celulare se obține determinând în mod corespunzător setul de operatori sau template [77,97,126].

Comparativ cu procesoarele numerice, care sunt relativ uşor de programat, determinarea sau proiectarea setului de operatori pentru o rețea neuronală celulară, care să asigure o anumită prelucrare dorită, este mult mai dificilă [118]. Se confirmă şi în cazul rețelelor neuronale celulare faptul că, dacă se doreşte ca rețeaua să lucreze rapid, în timp real, trebuie să se "consume" mult timp cu antrenarea sau învățarea ei. Acest fapt este lesne de justificat dacă avem în vedere că o rețea de dimensiune 100*100 de celule constituie un bloc analogic cu 10.000 celule interconectate. Dinamica acestei rețele trebuie astfel "acordată" încât pe baza ecuației diferențiale de stare asociată rețelei să realizeze tocmai prelucrarea dorită asupra imaginii de intrare şi de stare. În cazul operatorilor 3*3, sunt 19 parametri care trebuie determinați (2*9+1).

La proiectarea operatorilor se au în vedere două condiții deosebit de importante din punct de vedere al posibilității de implementare pe chip a rețelei neuronale celulare:

1.) În aplicațiile curente se folosesc aproape în exclusivitate rețea cu operatori liniari, de dimensiune 3*3, deoarece deocamdată numai acestea pot fi implementate pe circuitele CNN existente. Sunt elaborate metode pentru a se putea implementa şi operatori cu dimensiunea 5*5 sau cu neliniarități simple, prin transformarea lor în operatori liniari, de dimensiune 3*3 [59].

2.) În cursul proiectării se va avea în vedere faptul că datorită imperfecțiunilor tehnologice chiar dacă operatorii sunt considerați invarianți în spațiu, există o dispersie spațială a valorilor acestora. De aceea, dacă este posibil se va alege acea variantă care va asigura o funcționare cât mai robustă a rețelei, adică cât mai tolerantă la erorile datorate imperfecțiunilor tehnologice [24,74,118]. După determinarea unui operator şi testarea funcționării lui corecte prin simulare are loc etapa optimizării (acordării) valorilor elementelor operatorilor, în vederea funcționării corecte a operatorului şi la implementarea pe chip-CNN. Valorile optime astfel obținute pot fi dependente de familia de chip-CNN utilizat.

Pentru proiectarea operatorilor CNN se pot utiliza următoarele metode:

- metoda intuitivă, euristică sau empirică;
- metoda învățării;
- metoda proiectării directe.

Prima metodă se bazează în mod esențial pe intuiția proiectantului şi se poate aplica cu succes numai în cazuri simple existând însă şi situații când nu se poate garanta găsirea operatorilor doriți. Proiectantul trebuie să aibă experiență mare în procesarea imaginilor şi în domeniul CNN, pentru a "specula" soluția favorabilă.

Asemănător rețelelor neuronale clasice, metoda învățării se poate utiliza şi în cazul rețelelor neuronale celulare. Determinarea unor metode performante de antrenare pentru aceste rețele reprezintă în momentul de față un domeniu de cercetare de certă actualitate [5].

Utilizarea metodei de învățare în cazul rețelelor neuronale celulare, impune luarea în considerare a anumitor particularități ale acestora.

Metoda învățării supervizate se bazează pe existența unor perechi de intrare şi de ieşire dorită, astfel încât să se obțină operatori din ce în ce mai buni în decursul antrenamentului. Multe metode de învățare a rețelelor neuronale folosesc principiul evaluării într-un punct din spațiul parametrilor a unor funcții de cost. În cadrul procesului de învățare se caută minimizarea valorii acestor funcții de cost. Ca metodă de minimizare se poate utiliza metoda gradientului.

Găsirea minimului funcției de cost pe cale analitică este o metodă utilă şi rapidă, dacă se poate identifica o funcție de cost potrivită [37,39].

Practic metodele diferă una de alta, prin criteriul după care se trece de la un punct la altul, în cadrul metodei iterative de minimizare a funcției de cost.

Dintre algoritmii de învățare cunoscuți şi aplicați în general la rețele neuronale, în cazul rețelelor neuronale celulare se foloseşte, mai cu seamă algoritmul genetic.

Algoritmul genetic [61] fiind un model numeric al evoluției, imită selecția naturală. Evoluția şi selecția naturală au un rol important în biologie, dar îşi găsesc aplicație şi în cadrul inteligenței artificiale. Algoritmul genetic se poate considera ca şi o metodă stohastică de căutare.

Algoritmul genetic, în contrast cu alte metode, nu studiază punctele singulare ale funcției de cost din spațiul parametrilor, ci se ocupă la un moment dat cu mai multe puncte, lucrând cu populații. Elementele populației vor determina diferite valori ale funcției de cost, unele vor fi mai apropiate de minimul global, pentru care valoarea

funcției de cost este mai mică, dar vor exista şi elemente ale populației care vor determina valori mai îndepărtate de acest minim global. Astfel, elementele populației vor avea proprietăți diferite. Se doreşte ca algoritmul genetic să realizeze o selecție spontană, asemănătoare selecției naturale, astfel încât în cadrul evoluției să se obțină populații care să aibă cât mai multe elemente situate aproape de minimul global şi cât mai puține elemente aflate la distanță mare față de punctul care asigură minimul global al funcției de cost. La aplicarea algoritmului genetic în cazul rețelelor neuronale celulare, problema deosebită o reprezintă determinarea unui mod de codare a elementelor operatorilor şi găsirea formei de exprimare adecvate pentru funcția de cost [61].

Pentru rețelele neuronale celulare, în foarte multe cazuri însă, operatorul sau lucrează corect sau nu lucrează deloc, fără stări intermediare, fiind astfel dificilă determinarea unor funcții de cost sau de energie. O altă particularitate a rețelelor neuronale celulare o constituie faptul că pentru multe cazuri complexe nu există un singur operator invariant în spațiu şi timp, astfel încât procesul de învățare nu se va opri niciodată cu un rezultat acceptabil.

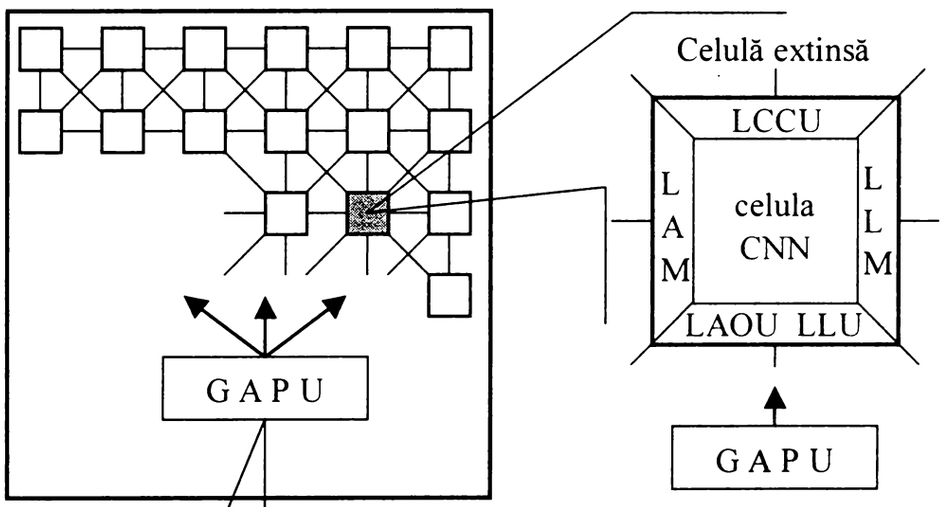
Eficiența utilizării metodei de învățare pentru determinarea operatorilor s-a demonstrat deocamdată numai la procesarea de imagini simple binare; în multe alte aplicații operatorii se pot obține mai uşor prin metoda proiectării directe.

Cea de a treia metodă de proiectare de operatori este metoda proiectării directe. Această metodă se poate aplica mai ales în cazul imaginilor binare, la rețele cuplate înainte şi cuplate înapoi, la care se cunoaşte exact funcția dorită [118]. Metoda este dependentă de particularitățile operatorilor. Comparativ cu metodele anterioare care oferă un număr finit de soluții pentru operatorii doriți, proiectarea directă furnizează toate variantele posibile dintre care se va alege, evident, cea mai robustă. Un alt avantaj al acestei metode este că necesită o putere de calcul mult inferioară față de metoda învățării.

Pentru prelucrarea imaginilor cu niveluri de gri utilizând rețele neuronale celulare, un grad sporit de dificultate îl prezintă determinarea operatorilor de reacție de tip A.

2.10 Calculatorul universal CNN

Calculatorul universal CNN (CNN Universal Machine [87]), s-a obținut prin generalizarea noțiunii CNN. La baza calculatorului universal CNN stă procesorul analogic sau chipul CNN. Procesorul analogic realizează un program analogic cu ajutorul unei memorii locale pe baza unui program format din instrucțiuni analogice și logice, adică prin calcul dual. Semnalele cu care lucrează microprocesorul analogic pot fi analogice și numerice și la fel și operatorii utilizați pot realiza funcții analogice sau operații logice. Un program CNN analogic conține algoritmi care se realizează atât secvențial cât și paralel. CNN-UM este primul calculator analogic programabil, cu limbaj de programare și sistem de operare, care are puterea de calcul a unui supercomputer pe un singur chip. În figura 2.14 este prezentată structura unui calculator universal CNN.



GAPU	unitatea de programare analogică globală
APR	registrul global al programului analogic
LPR	registrul global al programului logic
SCR	registrul global de control a comutărilor
GACU	unitatea analogică globală de control

Figura 2.14. Structura calculatorului universal CNN-UM.

Comanda, în ansamblu, a procesorului este realizată de către unitatea de programare analogică globală (GAPU - Global Analogic Programing Unit). Această

unitate conține registrul global al programului analogic cu operatori (APR - Global Analog Program Register) şi registrul global cu instrucțiuni logice (LPR - Global Logic Program Register). Realizarea corectă a interacțiunilor între celulele rețelei se obține cu ajutorul registrului care comandă conexiunile locale (SCR - Switch Configuration Register). Unitatea globală de control analogic (GACU - Global Analogic Control Unit) conține programul de comandă a instrucțiunilor analogice în cod maşină. Programul de comandă a instrucțiunilor analogice în cod maşină este rezultatul unei compilări al unui program analogic de nivel înalt sau a unui interpretor. O celulă extinsă conține în afară de elementele specifice celulei propriu-zise şi alte elemente: unitatea logică locală (LLU – Local Logic Unit), unitatea locală analogică de ieşire (LAOU - Local Analog Output Unit), unitate locală de memorie analogică cu mai multe elemente de memorie analogică locală (LAM - Local Analogic Memory), memorie logică locală (LLM - Local Logic Memory) şi unitate locală de control şi comunicare a celulei (LCCU - Local Comunication and Control Unit).

CNN-UM realizează programe cu algoritmi analogici, prin instrucțiuni cu operatori analogici şi operații logice în cazul în care se utilizează imagini binare. Pentru realizarea unor astfel de programe este nevoie de imaginea originală, de imagini cu rezultate parțiale, de memorarea unor operatori logici şi memorarea unor operatori analogici.

Memoriile analogice locale (LAM) servesc la memorarea imaginilor de intrare, de stare şi de ieşire, imagini cu niveluri de gri. Aceste memorii sunt foarte importante deoarece existența lor asigură memorarea imaginilor intermediare pe chip, care vor fi supuse unor prelucrări cu alți operatori, conform programului analogic general. Unitatea logică, cu două intrări şi o ieşire, este programabilă şi poate realiza o funcție logică pentru pixelii din aceaşi poziție a două imagini binare. În acest caz nu se realizează nici o conexiune logică cu celulele vecine. Pentru imagini binare se folosesc memoriile LLM. Scrierea şi citirea din memoriile LAM şi LLM se poate face pentru o linie întreagă, utilizând valori analogice sau binare, astfel încât transferul pe o magistrală M dimensională a unei imagini complete M*N se poate face în N paşi; se micşorează astfel substanțial timpul necesar pentru încărcarea şi extragerea imaginilor de pe chip. Reducerea efectivă a timpului de lucru se va asigura prin procesarea paralelă.

2.11 Mediul de dezvoltare CNN "CadetWin"

Proiectarea, testarea şi optimizarea aplicaţiilor CNN este posibilă numai dacă există un mediu de dezvoltare CNN adecvat. Această unealtă reprezintă o variantă situată între două limite: simulare 100% soft şi emularea hard. Pentru simularea reţelelor neuronale celulare se poate utiliza mediul de programare MATLAB [40], pentru care a fost elaborat un "toolbox" adecvat, ce poate fi utilizat atât pentru reţele analogice [85] cât şi pentru DTCNN [29].

Interesul deosebit pe plan mondial acordat cercetării în domeniul reţelelor neuronale are la bază posibilitatea realizării circuitelor CNN în tehnologie VLSI. Aceste chipuri, pentru a putea fi practic utilizate, trebuie să fie însoţite de un mediu de dezvoltare de aplicaţii. În cadrul prezentei teze de doctorat a fost utilizat mediul de dezvoltare "CadetWin" (CNN application development environment and toolkit under Windows [123]), care are următoarele componente:

1.) Platforma "VisMouse" (CNN visual mouse software platform for Windows [124,127]) bazat pe concepţia "visual mouse". Acesta este un mediu software integrat pentru proiectarea de aplicaţii CNN. Cu ajutorul acestei platforme se pot achiziţiona, memora şi vizualiza imagini de intrare şi imagini de ieşire, se poate comanda camera video precum şi alte accesorii. De pe această platformă se pot accesa şi alte mijloace software ale mediului de dezvoltare CadetWin.

2.) "TemMaster" (Template design and optimization tool for binary input-output CNN's [126]) este un software dedicat pentru proiectarea şi optimizarea de operatori.

3.) "SimCNN" (Multi-layer CNN simulator for visual mouse platform [128]) este un program de simulare cu ajutorul căruia se pot testa operatori respectiv se pot proiecta, testa şi optimiza algoritmi ori subrutine spaţiio-temporale care includ operaţii aritmetice şi logice şi care utilizează reţele neuronale celulare monostrat sau multistrat.

4.) Editorul şi compilatorul "Alpha" (CNN Alpha language and compiler [129) care permite editarea şi compilarea programelor într-un limbaj de nivel înalt specific CNN.

5.) Editorul şi interpretorul în limbaj de nivel inferior (asamblor) "AMC" (Extended analogic macro code and interpreter [130]).

6.) Editor şi interpretor de programe pseudocod în "Script" [124,128].

7.) Interfaţa "CCPS" (CNN chip prototyping system [131]), dintre calculator şi platforma cu chip-CNN. Cu ajutorul acestei interfeţe se poate conecta şi aborda în mod unitar oricare platformă cu chip-CNN nou în vederea testării şi perfecţionării ei.

8.) Programarea în mediu CNN este facilitată de existenţa unei biblioteci software de operatori şi algoritmi care se dezvoltă şi se revizuieşte periodic pe baza rezultatelor cercetărilor CNN din întreaga lume [125].

Prin utilizarea mediului de dezvoltare CadetWin se pot realiza programe CNN în limbaj de nivel înalt specific Alpha sau în limbaj de asamblare AMC. Este important de semnalat faptul că aceste programe sunt unitare şi compatibile sub aspectul rulării în mediul CadetWin atât pe mijloace de simulare software sau emulare hardware cât şi pe chipul CNN analogic.

În figura 2.15 se prezintă etapele de realizare ale unui algoritm analogic, de la proiectare până la simulare soft, emulare hardware respectiv testare direct pe un chip CNN [92].

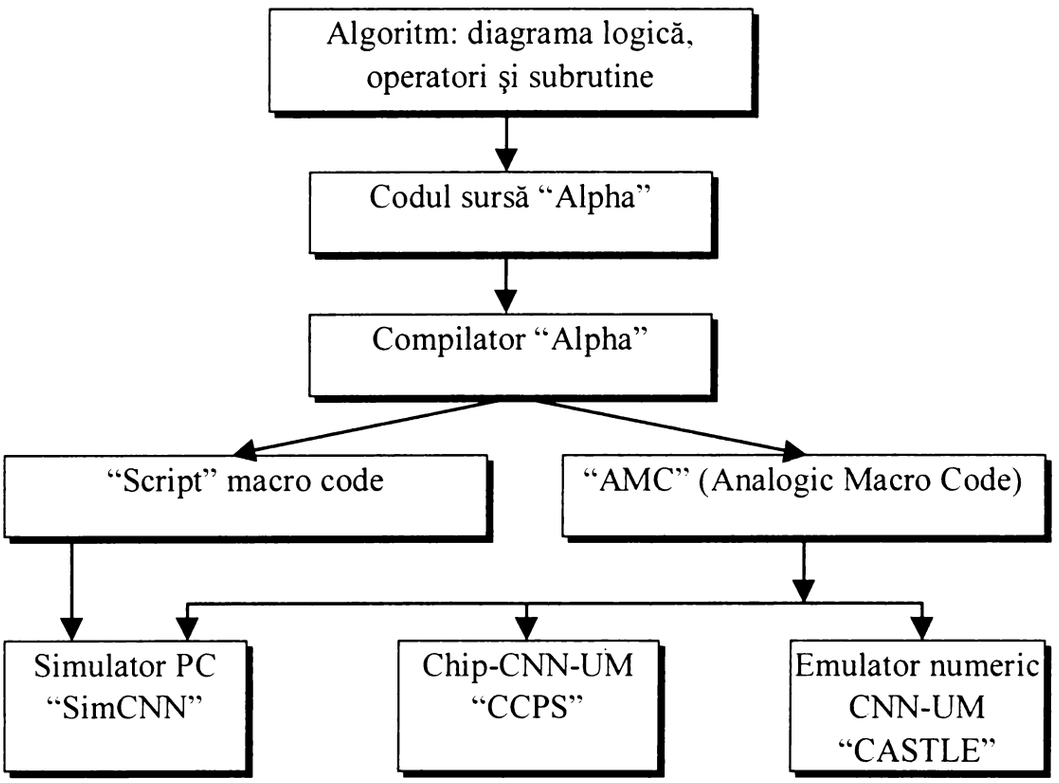


Figura 2.15. Etapele de elaborare ale unui algoritm analogic CNN.

2.12 Circuite integrate CNN

În scurta perioadă de timp, de când au fost propuse prima oară rețelele neuronale celulare, în laboratoarele de cercetare din întreaga lume s-au proiectat şi realizat peste 10 prototipuri de chip-CNN analogice [21,68,78] şi circuite de emulare numerice [57].

Dintre aceste realizări se prezintă două tipuri de circuite CNN utilizate în experimentele prezentate în această lucrare.

1.) Circuitul CNN cP 400 [21].

Circuitul CNN cP 400 este primul procesor analogic care prezintă caracteristicile specifice unui calculator CNN-UM. Cu ajutorul lui poate fi executat un program analogic care include operații analogice şi logice utilizând imagini binare din memoria locală. Acest circuit are următoarele caracteristici:

- rețeaua neuronală este formată din 20*22 de celule;
- procesează imagini binare cu valori standard CNN, adică -1 pentru nivelul de alb şi +1 pentru nivelul de negru;
- ecuația de stare care caracterizează funcționarea unei celule este dată de relația:

$$\frac{dx^c}{dt} = -g(x^c) + \sum A_d x^d + \sum B_d u^d + D_A + D_B, \qquad (2.49)$$

unde x^c şi u^c reprezintă starea respectiv intrarea celulei "c", iar x^d şi u^d semnifică starea respectiv intrarea corespunzătoare unei celule din rețea, din vecinătatea celulei "d". Variabilele D_A şi D_B precum şi polarizările, sunt codate pe 8 biți, dintre care unul este pentru semn;

- caracteristica funcției de transfer neliniare g(x) este prezentată în figura 2.16;

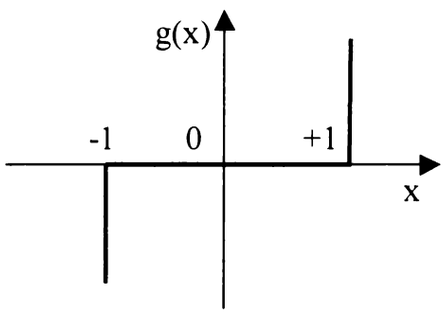


Figura 2.16. Caracteristica de transfer g(x).

- la o procesare elementară în condiții optime, constanta de timp caracteristică sau timpul de stabilizare pentru o celulă este τ_{CNN}=250 ns;
- se pot memora local cel mult 4 imagini binare, care pot fi imagini de intrare de la senzorul optic sau de pe platformă, ori imagini de ieşire rezultate în urma unor prelucrări anterioare. La achiziția imaginilor pragul de binarizare este reglabil;
- pot fi memorate local cel mult 8 seturi de operatori (A, B, z), cu cel mult 20 de elemente reprezentate fiecare pe câte 8 biți. Dimensiunea unui operator este 3*3. Valorile elementelor în modul nu pot depăşi valoarea 3 în cazul operatorilor de tip A şi B, iar în cazul curentului această valoare este 6.

2.) Circuitul analogic CNN cP 4000 [68].

Caracteristicile acestui chip-CNN analogic sunt mai mult decât promițătoare. Cele mai semnificative proprietăți sunt după cum urmează:

- rețeaua este formată din 64*64 de celule;
- utilizează imagini cu niveluri de gri şi imagini binare cu valori standard CNN, (adică este atribuit domeniul de valori [-1,1] pentru nivelurile situate între alb şi negru);
- pot fi memorate pe chip 32 de seturi de operatori (A, B, z);
- poate memora local în LAM cel mult 4 imagini cu niveluri de gri precum şi alte 4 imagini binare în LLM;
- prezintă facilitatea utilizări imaginii mască binară.

Avantajul major al procesării paralele prin utilizarea rețelelor neuronale celulare este prezent însă numai în cazul implementării acestora pe un chip. De aceea, proiectarea şi realizarea de chipuri CNN cât mai performant este o direcție de cercetare pentru care se depun în continuare eforturi umane şi materiale substanțiale. Astfel, în anul 1999 s-a proiectat o nouă variantă de circuit CNN cu dimensiunea de 176*144 celule, cu intrări şi ieşiri binare [79].

3. Algoritm analogic CNN pentru urmărirea obiectelor aflate în mişcare

În acest capitol este prezentat un algoritm CNN original, bazat pe imagini de urmărire a unui obiect în mişcare cu ajutorul unei camere video montate pe braţul unui robot cu două grade de libertate. Între două imagini de intrare consecutiv achiziţionate, prin prelucrarea complet paralelă cu reţele neuronale celulare a semnalelor bidimensionale, se asigură efectuarea tuturor procesărilor necesare în vederea funcţionării în timp real a sistemului.

Totodată, este prezentat şi un nou algoritm CNN, propus de autor, pentru determinarea punctului central al unui obiect utilizând reţele neuronale celulare, care se bazează numai pe procesări elementare logice şi cu operatori liniari de dimensiune 3*3, fiind astfel posibilă implementarea cu uşurinţă a acestuia pe un chip-CNN de 64*64 de celule (cP4000 [68]). În acest fel, prelucrarea se poate efectua chiar în timp real, datorită procesării complet paralele şi prin reducerea substanţială a numărului de încărcări pe chip a imaginilor iniţiale.

3.1 Comanda vizuală a unui robot

Extinderea domeniilor de aplicare a roboţilor prin creşterea adaptibilităţii acestora, ca urmare a integrării senzorilor în structura robotului reprezintă actualmente o problemă fundamentală. Dintre senzorii utilizaţi în robotică, senzorul vizual sau camera video ocupă un loc aparte, deoarece utilizarea lui permite evaluarea fără contact a spaţiului de lucru.

Prin includerea senzorului vizual într-o buclă de reacţie în care este procesată informaţia conţinută în imagini se poate controla poziţia efectorului final sau ghida un robot mobil, spre un obiect ţintă. De cele mai multe ori extragerea informaţiei din imagine, "priveşte" (looking) şi comanda efectivă a poziţiei, adică "mută" (moving), sunt incluse într-aşa zisă buclă deschisă. În consecinţă, precizia operaţiilor realizate depinde direct de acurateţea senzorului vizual de precizia de poziţionare a efectorului final al robotului.

Pentru mărirea preciziei, senzorul vizual se include într-o buclă cu reacție negativă vizuală, denumită în general comandă vizuală (visual servoing sau visual servo) [54,110,112].

Comanda vizuală a roboților presupune extragerea de trăsături din imaginile achiziționate, pe baza cărora se realizează modelele obiectelor din imagine. De cele mai multe ori aceste trăsături caracterizează puncte care rezultă în urma proiecției unor obiecte în planul de imagine al camerei video realizându-se modelul geometric al obiectelor. Aceste puncte au ca şi parametri coordonatele față de un sistem de referință ataşat planului de imagine.

Fiind dat un set de k parametri ai unei trăsături din imagine, se poate defini vectorul parametru al acestei trăsături din imagine $\mathbf{f}=[f_1, f_2,\ldots, f_k]^T$. Deoarece oricare element al lui $\mathbf{f}$ este un parametru real rezultă că $\mathbf{f}\in\mathbf{F}\subseteq\mathbf{R}^k$, unde $\mathbf{F}$ este spațiul parametrilor trăsăturii din imagine. De exemplu, dacă se utilizează transformarea în perspectivă pentru proiecția unui punct oarecare P din spațiu, pe planul imaginii, trăsătura $\mathbf{f}$ este chiar vectorul de poziție în sistemul de referință ataşat acestui plan, adică $\mathbf{r}=[x, y]^T$, iar $\mathbf{F}\subseteq\mathbf{R}^2$ este spațiul coordonatelor x şi y din planul imaginii.

Pentru comanda vizuală a unui robot se foloseşte una din următoarele configurații de amplasare a camerei video:

1.) Camera video este fixată de efectorul final (figura 3.1a), configurația cunoscută sub denumirea "ochi în mâna" (eye in hand), [54,80,107,112]. Față de sistemul de referință ataşat de baza robotului, poziția punctului țintă P este descrisă de vectorul $\mathbf{r_t}$, iar vectorul de poziție corespunzător efectorului final este $\mathbf{r_e}$. Există o relație cunoscută ${}^{e}\mathbf{r_c}$, deseori constantă, între poziția camerei ori a camerelor video şi poziția efectorului final. Vectorul de poziție al punctului țintă față de cameră este ${}^{c}\mathbf{r_t}$.

2.) Camera video are o poziție fixă, situată în spațiul de lucru, ca în figura 3.1b [2,4,58,70]. Poziția camerei (sau a camerelor video) $\mathbf{r_c}$ este constantă față de sistemul de referință ataşat de baza robotului. În acest caz imaginea țintei respectiv vectorul de poziție ${}^{c}\mathbf{r_t}$, sunt independente de mişcarea robotului.

Există şi combinații a celor două structuri prezentate mai sus; de exemplu, cazul în care se urmăreşte poziția efectorului final al unui robot cu o cameră video montată pe efectorul final al unui alt robot.

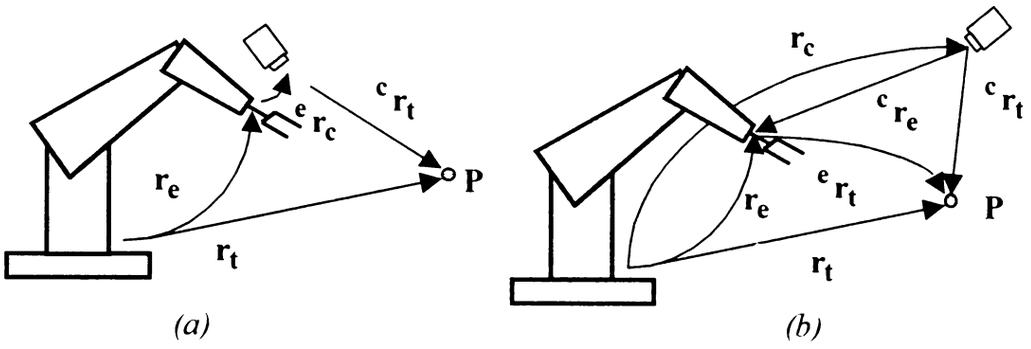


Figura 3.1. Configurații de amplasare a camerei video la comanda vizuală a unui robot:(a) camera video este fixată de efectorul final; (b) camera video fixă.

Sistemele de comandă vizuală se pot clasifica pe baza a două criterii [50,54].

1.) După structura sistemului de comandă a robotului:

1.a) Structură dinamică "priveşte şi mută" [54,112].

În acest caz, în structura sistemului de comandă a robotului sistemul vizual este în ordinea ierarhică superioară, adică sistemul vizual furnizează mărimile de referință pentru sistemul de conducere al cuplelor cinematice. Această configurație foloseşte traductoarele interne pentru atingerea poziției prescrise a robotului.

1.b) Comandă vizuală "directă" [50,58].

În cazul acestei structuri sistemul vizual comandă direct cuplele cinematice, astfel încât nu există altă buclă de reacție decât cea vizuală pentru stabilirea poziției finale a mecanismului robotic.

2.) După modul de definire a erorii de poziționare:

2.a) Comandă vizuală bazată pe poziții [75,80,112].

Semnalul eroare este definit în coordonatele aferente spațiului de lucru tridimensional. Identificarea poziției țintă se realizează pe baza unui model geometric al obiectului țintă, la nivelul planului de imagine ataşat camerei video, prin extragerea de trăsături care sunt utilizate pentru estimarea poziției țintei în funcție de poziția camerei video. Reacția negativă minimineză eroarea în spațiul pozițiilor estimate. Avantajul major al metodei de comandă vizuală bazată pe poziții este faptul că sarcinile de comandă se pot descrie în coordonatele carteziene, utilizate frecvent în robotică. Dezavantajul metodei este legat de faptul că mărimea de reacție este determinată aproximativ, fiind dependentă de parametrii de calibrare ai sistemului. În consecință, comanda vizuală bazată pe poziții este extrem de sensibilă la erorile de calibrare. Înaintea utilizării efective a informației vizuale pentru comanda unui robot este necesară calibrarea camerei video. Această operație constă în [54,122]:

- determinarea parametrilor intriseci ai camerei, cum ar fi de exemplu distanța focală;
- determinarea poziției camerei video (${}^{e}\mathbf{r}_{c}$) față de poziția efectorului final sau calibrarea "ochi mână" (figura 3.1a).
- determinarea, pentru camera fixă, a poziției față de sistemul de referință atașat de baza robotului, $\mathbf{r}_{c}$ (figura 3.1b);

Un alt dezavantaj al acestei metode de comandă este necesitatea realizării unui model clar al obiectului țintă.

2.b) Comandă vizuală bazată pe imagini [4,50,58,107].

Sarcina de comandă vizuală bazată pe imagini poate fi reprezentată printr-o funcție de eroare de imagine definită direct funcție de parametrii trăsăturii imaginii:

$e:F\rightarrow\mathbf{R}^{l}$, unde $l \leq k$, k fiind dimensiunea spațiului parametrilor trăsăturilor de imagine. Pentru o sarcină de comandă realizată e=0.

Chiar dacă se utilizează un sistem "ochi în mână" greșit calibrat, dacă reacția sistemului este asimptotic stabilă, eroarea de imagine va tinde spre zero şi eroarea cinematică va tinde la zero. Astfel, unul din avantajele principale ale comenzii bazate pe imagini față de comanda bazată pe poziții este că precizia poziționării sistemului este mai puțin sensibilă la erorile de calibrare ale camerei video. Un alt avantaj îl reprezintă posibilitatea reducerii timpului de procesare.

Pe baza acestor criterii de clasificare rezultă cele 4 structuri de bază pentru comanda vizuală a unui robot, prezentate în figurile 3.2-3.5. Indiferent de structura utilizată, sistemul vizual trebuie să extragă informația necesară pentru a putea îndeplini sarcinile de comandă urmărite.

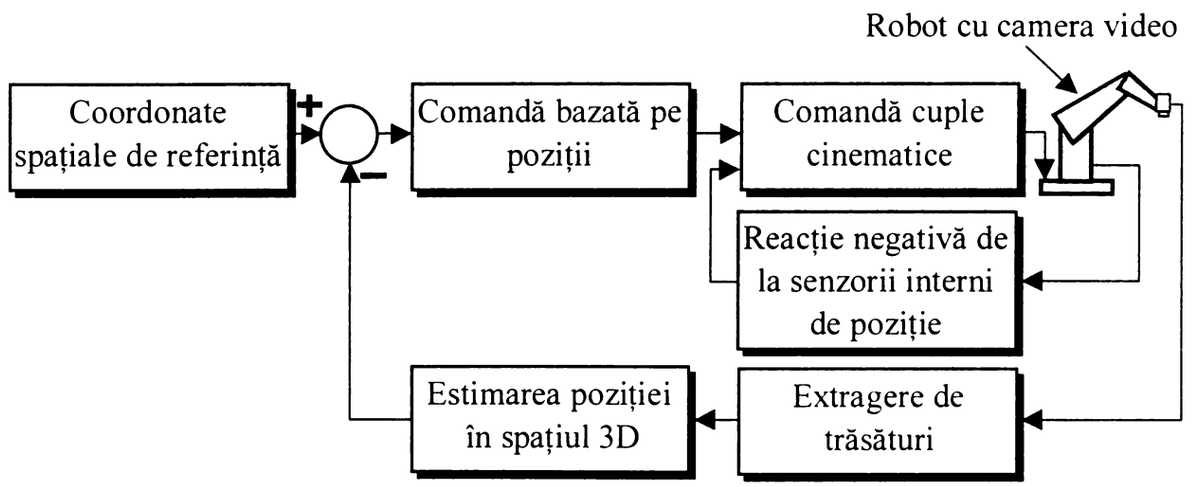


Figura 3.2. Structură dinamică "priveşte şi mută" bazată pe poziții.

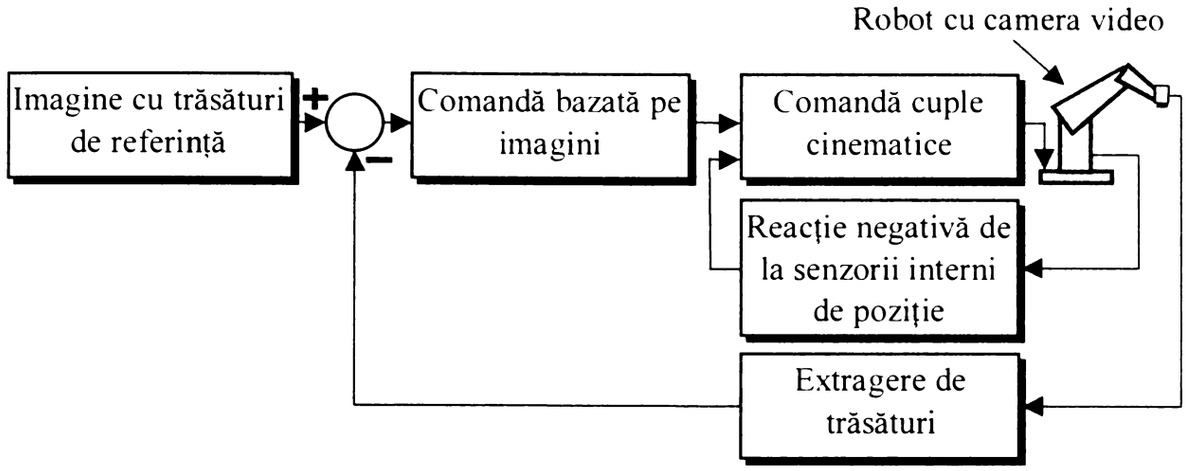


Figura 3.3. Structură dinamică "privește și mută" bazată pe imagini.

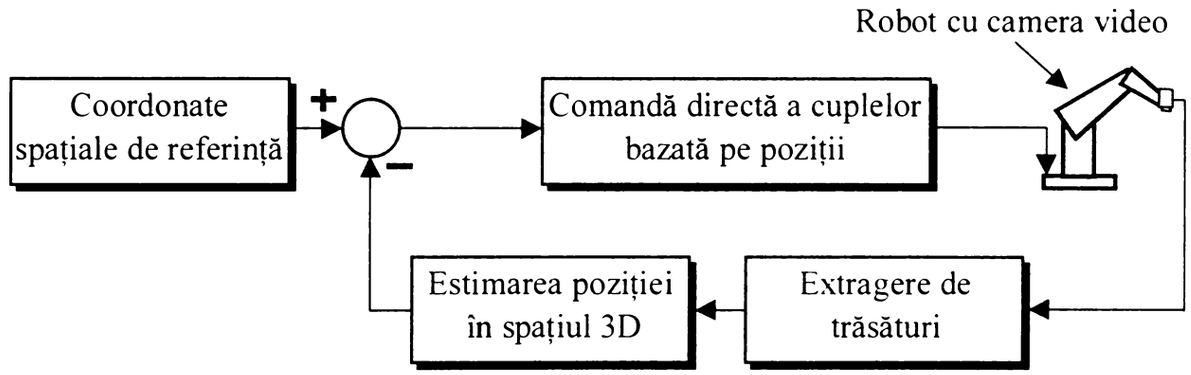


Figura 3.4. Comandă vizuală "directă" bazată pe poziții.

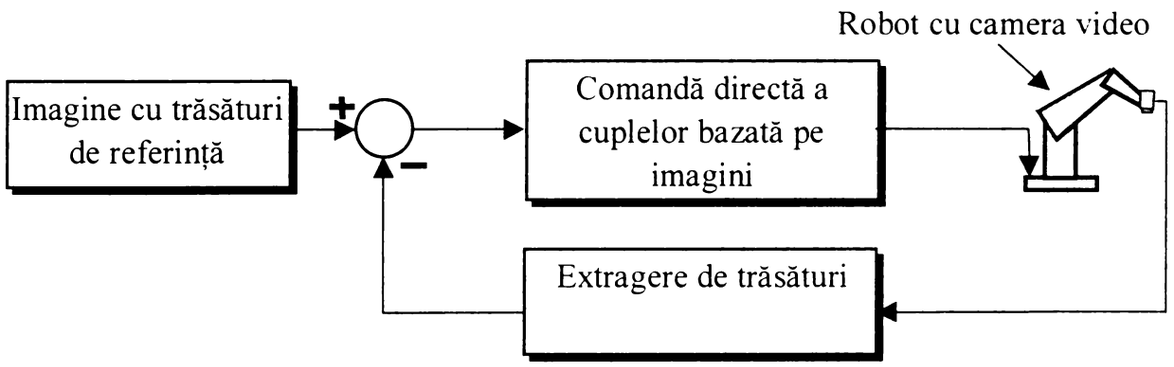


Figura 3.5. Comandă vizuală "directă" bazată pe imagini.

Aproape toate sistemele utilizate practic adoptă structura dinamică "privește și mută", datorită următoarelor motive:

i.) Rata de achiziție și prelucrare pentru imagini este mai mică față de rata cu care se pot achiziționa și prelucra semnalele traductoarelor interne ale robotului.

ii.) Această structură permite separarea sistemului vizual față de sistemul conducere al robotului, considerat ideal în cele ce urmează.

iii.) Majoritatea roboților au o interfață standard care permite comanda în poziție şi viteză. Astfel, sistemul vizual se poate conecta relativ uşor la sistemul de conducere al robotului.

Sistemele de comandă vizuală care observă doar obiectul țintă sunt sisteme în buclă deschisă a punctului final "EOL"(end point open-loop), iar cele care au în observare directă şi efectorul final şi ținta sunt sisteme în buclă închisă "ECL"(end point closed-loop).

O aplicație tipică a comenzii vizuale directe este poziționarea unui efector final cu o cameră montată pe el în aşa fel încât să prehenseze obiectele. Poziția efectorului final față de poziția obiectului se determină indirect prin relația cinematică cunoscută dintre acesta şi poziția camerei video [2,80,112]. Erorile din această relație cinematică conduc la erori de poziționare care nu pot fi corectate de sistem. Aceste erori pot fi eliminate dacă comanda vizuală a brațului de robot este bazată pe imagini sau prin observarea directă atât a poziției efectorului final cât şi a poziției țintei.

În acest capitolul se prezintă un algoritm CNN propus de autor, de urmărire a unui obiect în mişcare cu ajutorul unei camere video montată pe un braț robotic, utilizând comanda vizuală bazată pe imagini.

Problema este reluată în capitolul 5 al tezei, în care se propune un algoritm CNN de planificare a traiectoriei unui robot mobil care se deplasează autonom într-un mediu cu obstacole. Planificarea traiectoriei se face pe baza imaginilor furnizate de o cameră video fixă, care vizează atât robotul cât şi ținta.

3.2 Sistemul robotic utilizat la urmărirea obiectelor aflate în mişcare

Se va considera în cele ce urmează modelul simplificat al unui sistem robotic prezentat schematic în figura 3.6. Sistemul include un robot cu două grade de libertate şi un senzor vizual montat pe braţul unui robot. Robotul are numai cuple cinematice de rotaţie notate cu A şi B şi a căror poziţii sunt caracterizate prin unghiurile θ1 şi θ2.

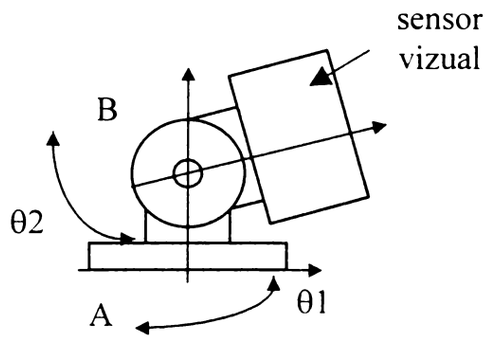


Figura 3.6. Sistem robotic cu comandă vizuală bazată pe imagini.

Se ataşează planului de imagine al senzorului vizual un sistem de referinţă, xOy, cu originea situată chiar în centrul imaginii. În planul imaginii furnizate de senzorul vizual se selectează un obiect şi se consideră că poziţia obiectului în plan va fi descrisă de coordonatele x_0 şi y_0 ale unui singur punct al obiectului numit punct central sau punct caracteristic (figura 3.7).

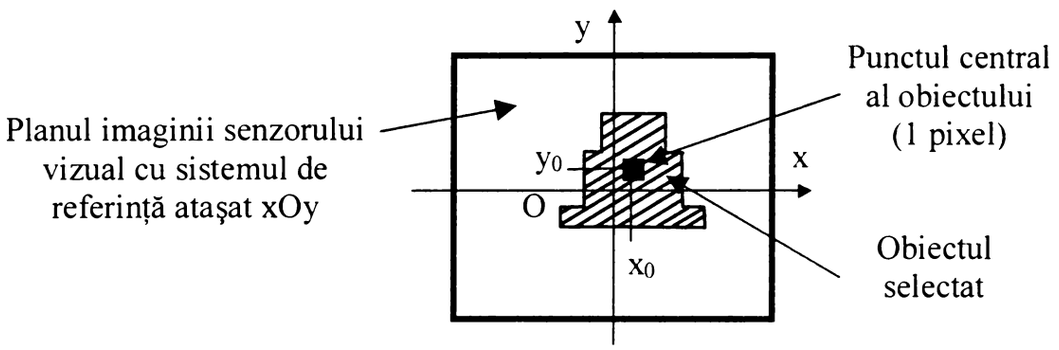


Figura 3.7. Precizarea poziţiei obiectului în planul imaginii senzorului vizual.

Punctul central al obiectului este punctul situat la jumătatea distanţei faţă de cele mai îndepărtate puncte ale obiectului, atât după direcţia x cât şi după direcţia y. În cazul în care obiectul prezintă axe de simetrie după direcţia x şi după direcţia y, punctul

central se găseşte la intersecţia axelor de simetrie. În principiu, punctul central al unui obiect se poate obţine prin procesul de decojire simetrică succesivă a obiectului cu câte un pixel pe fiecare direcţie. Acest proces se desfăşoară până la obţinerea unui singur pixel, considerat a fi chiar punctul central al obiectului.

În această aplicaţie se presupune că la momentul iniţial t_0, punctul central sau cel puţin un punct oarecare al obiectului desemnat este situat în originea sistemului de referinţă ataşat planului de imagine. Această condiţie se va realiza practic modificând, spre exemplu, manual, valorile iniţiale ale unghiurilor cuplelor cinematice, la valorile de referinţă $\theta 1_0$ şi $\theta 2_0$.

În principiu, poate fi utilizat orice alt criteriu de desemnare a obiectului, spre exemplu, desemnarea dintr-o imagine cu obiecte în repaus a acelui obiect care începe la un moment dat să se deplaseze. Algoritmul propus în acest capitol îşi păstrează şi în acest caz valabilitatea.

Sistemului robotic descris trebuie să asigure, ca după desemnarea unui obiect, indiferent de mişcarea acestuia în spaţiu, camera video să urmărească obiectul desemnat. În acest scop, poziţia braţului robotic trebuie astfel modificată încât la atingerea punctului de echilibru stabil al sistemului, punctul central al obiectului să se găsească în originea sistemului de referinţă xOy ataşat planului imaginii.

Obiectivul propus îl constituie creşterea vitezei de procesare în vederea asigurării funcţionării sistemului în timp real. În acest scop se va urmări utilizarea reţelelor neuronale celulare în cât mai multe etape de procesare din schema bloc a sistemului robotic prezentat. Astfel, începând de la achiziţia unei imagini curente şi până la determinarea unghiurilor de referinţă $\theta 1$ şi $\theta 2$, se doreşte ca procesarea să se realizeze numai în mediu CNN, pe acelaşi chip, în vederea micşorării timpului total de procesare. Imaginile de intrare se presupun achiziţionate cu pasul de eşantionare T. Funcţionarea în timp real impune ca între două eşantioane de imagini succesiv achiziţionate, să se realizeze toate prelucrările necesare pentru comanda robotului.

Dacă viteza de procesare este suficient de mare, chiar şi în cazul unor erori de poziţionare, sistemul de comandă al robotului va corecta poziţia curentă, înainte ca "să se piardă din privire" obiectul desemnat. Astfel, urmărirea obiectului se consideră continuă, chiar dacă algoritmul utilizează imagini achiziţionate la momente discrete de timp. Prin creşterea vitezei de procesare a semnalelor de intrare, poate rezulta posibilitatea urmăririi de obiecte care se deplasează rapid.

3.3 Algoritm pentru urmărirea obiectelor aflate în mişcare utilizând reţele neuronale celulare

În vederea obţinerii comportamentului dorit pentru sistemul robotic descris anterior, se propune algoritmul de comandă pe baza informaţiei vizuale prezentat în figura 3.8.

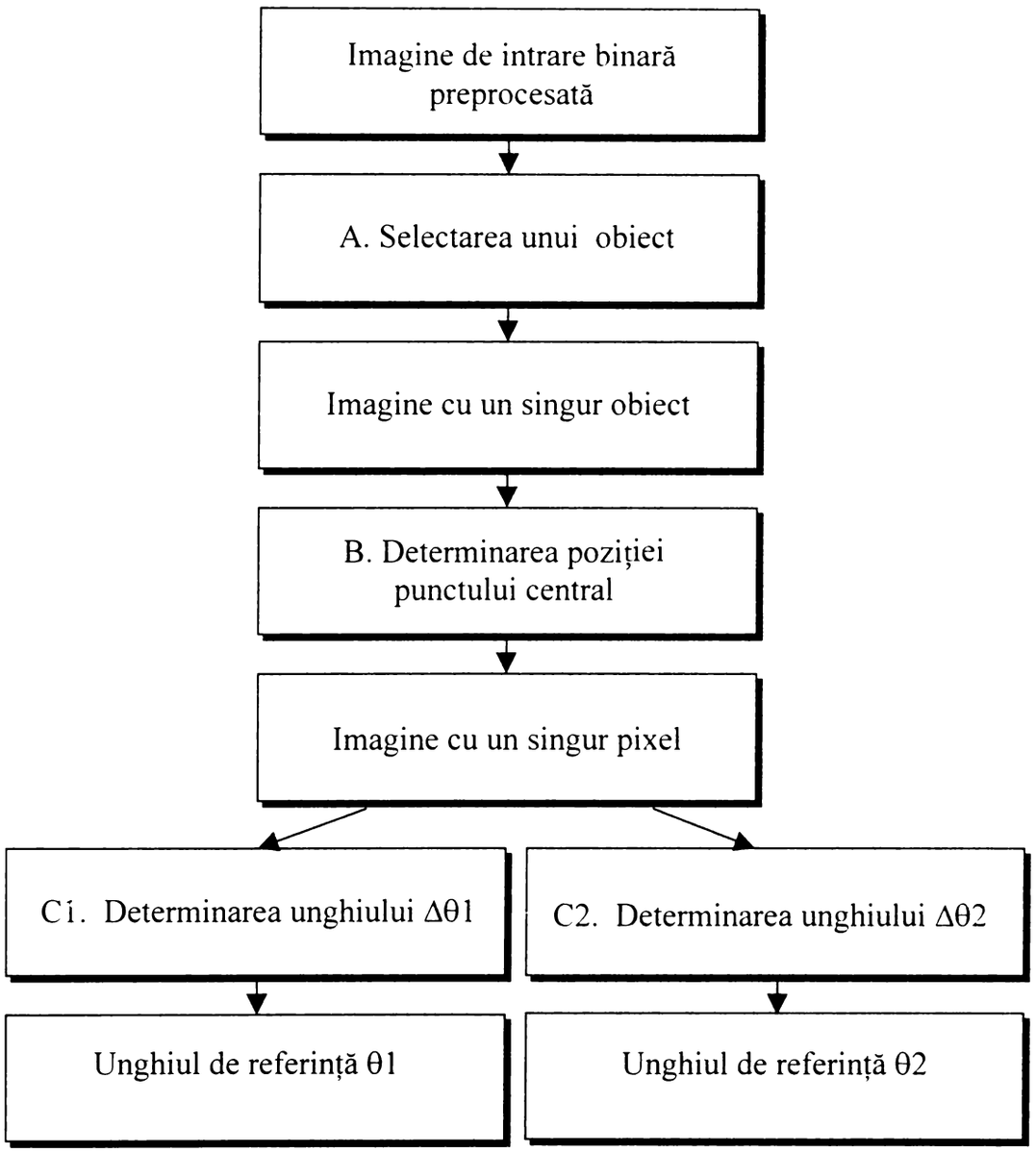


Figura 3.8. Organigrama algoritmului analogic CNN pentru urmărirea obiectelor aflate în mişcare.

Algoritmul pentru urmărirea obiectelor aflate în mişcare utilizează o imagine binară de intrare v(i,j) de dimensiune M*N. Imaginea binară se consideră ca fiind

rezultată în urma unor procesări elementare asupra imaginii achiziționate, cu niveluri de gri. În imaginea binară pixelii care aparțin obiectului au valoarea +1, iar celelalte elemente de imagine au valoarea –1.

$$v(i,j)=\begin{cases}+1 & \text{pentru pixelii care sunt în obiect}\\ -1 & \text{pentru pixelii care nu sunt în obiect}\end{cases}, \qquad (3.1)$$

pentru $1\le i\le M; 1\le j\le N$.

Acest algoritm cuprinde următoarele etape:

i.) Desemnarea la momentul inițial şi apoi selectarea automată a obiectului care va fi urmărit, la achiziția fiecărei imagini. Pentru desemnarea unui obiect în planul imaginii, la momentul inițial t_0, trebuie să se aducă cel puțin un punct al obiectului în originea sistemului de referință ataşat planului de imagine. După aceasta însă, la orice moment următor (t_0+kT) trebuie să se realizeze doar selectarea automată a obiectului desemnat la momentul t_0. Astfel, la sfârşitul acestei etape de procesare, imaginea de ieşire conține doar obiectul desemnat la t_0, dar în poziția curentă.

ii.) Determinarea în imaginea obținută după prima de procesare, a poziției P(x,y,kT) a punctului central a obiectului selectat. Algoritmul nu precizează coordonatele numerice ale punctului central în sistemul de referință ataşat planului de imagine ci realizează numai identificarea acestuia.

iii.) Determinarea variațiilor $\Delta\theta1$ şi $\Delta\theta2$ ale unghiurilor față de valorile actuale ale unghiurilor $\theta1$ şi $\theta2$ pentru cuplele cinematice de rotație ale brațului robotic. Unghiurile de referință noi ale cuplelor cinematice de rotație A şi B rezultă pe baza relațiilor:

$$\theta1_{nou}=\theta1_{actual}+\Delta\theta1, \qquad (3.2)$$

$$\theta2_{nou}=\theta2_{actual}+\Delta\theta2. \qquad (3.3)$$

Valorile unghiurilor $\Delta\theta1$ şi $\Delta\theta2$ se vor determina în aşa fel încât prin noua poziționare a brațului robotic (camerei video) punctul central al obiectul selectat din imaginea achiziționată să se situeze în originea sistemului de referință xOy, ataşat planului de imagine al senzorului vizual. În acest caz, prin poziționarea brațului robotic eroarea de imagine devine nulă şi se realizează sarcina de comandă care a fost propusă. Aceasta înseamnă de fapt găsirea unei soluții asemănătoare cu soluția pentru problema cinematică inversă a brațului sistemului robotic.

Pentru conducerea sistemului robotic pe baza informației vizuale "calibrarea" sistemului vizual cu sistemul de comandă rezultă din simpla suprapunere exactă a două imagini care au aceiași dimensiune, M*N.

Pentru implementarea algoritmului analogic CNN de urmărire a obiectelor aflate în mișcare se propune folosirea unei structuri de rețea neuronală celulară cu trei straturi, prezentată în figura 3.9.

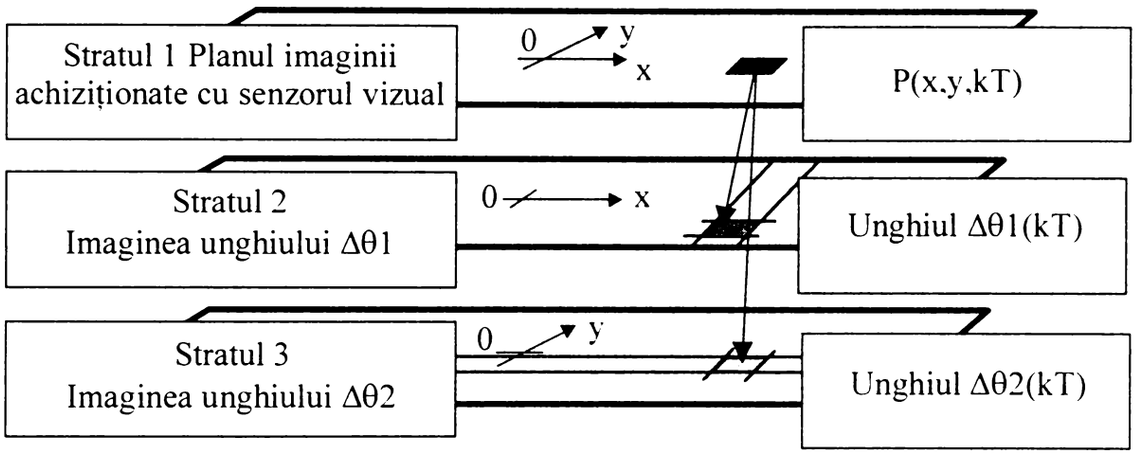


Figura 3.9. Structura de rețea neuronală celulară atașată comenzii sistemului

Stratul 1, superior, reprezintă chiar imaginea curentă de intrare. Prin selectarea unui obiect şi determinarea punctului central, în urma unor procesări elementare, va rezulta o imagine conținând un singur pixel activ, cu valoarea +1. Straturile 2 şi 3 sunt controlate de acest strat. Ieşirile celulelor din stratul 1 pot avea numai valori binare, adică +1 pentru pixelul activ (negru) şi –1 pentru pixelii inactivi (alb).

Stratul 2 se asociază cu "imaginea" Δθ1. Această imagine este necesară pentru determinarea unghiului de referință θ1, corespunzător fiecărei poziții pe orizontală a pixelului activ din stratul 1.

Stratul 3 se asociază cu "imaginea" Δθ2. Corespunzător fiecărei poziții pe verticală a pixelului activ din stratul 1, cu ajutorul imaginii Δθ2 se determină unghiul de referință θ2.

Straturile 2 şi 3 adică "imaginile" Δθ1 şi Δθ2 funcționează ca nişte memorii bidimensionale care sunt adresate în poziție de pixelul activ din stratul 1. Domeniul valorilor de la ieşirea celulelor (conținutul locațiilor de memorie) din straturile 2 şi 3 sunt cuprinse în intervalul [-1,+1], adică imaginile corespunzătoare sunt cu niveluri de gri. Imaginile corespunzătoare straturilor 1, 2 şi 3 au dimensiuni identice. Determinarea unghiurilor Δθ1 şi Δθ2 se face pe baza suprapunerii stratului 1 cu straturile 2 şi 3 şi selectarea a câte unei locații de memorie de la adresa indicată de pixelul activ din stratul

1. În acest fel este suficientă numai identificarea poziției locației prin intermediul imaginii stratului 1. Pentru o memorie bidimensională clasică determinarea unei valori pentru o locație de memorie este imposibilă dacă nu se cunosc numeric coordonatele spațiale ale locației.

Pentru fiecare etapă din algoritmul analogic CNN pentru urmărirea obiectelor aflate în mişcare se utilizează numai soluții care se pot implementa pe circuitele CNN, prezentate în capitolul 2. Semnalele care se utilizează, indiferent de natura lor, sunt interpretate numai ca imagini, toate prelucrările făcându-se în acest mediu compatibil CNN, până la obținerea noilor valori pentru unghiurile de referință ale cuplelor cinematice. Se doreşte ca fiecare etapă să se realizeze printr-o singură procesare adică cu un singur operator. Operatorii care se folosesc trebuie să fie liniari, cu dimensiunea 3*3, astfel încât să permită implementarea şi testarea pe un chip-CNN real. Acolo unde este posibil se va utiliza deci un singur operator cunoscut [125]. Dacă într-o anumită etapă dată, soluțiile oferite de operatorii existenți nu sunt performante, atunci se impune elaborarea de noi operatori şi algoritmi CNN.

În cazul implementării algoritmului de urmărire continuă a unui obiect pe un chip-CNN programabil, pot să apară problemele specifice de adaptare a operatorilor algoritmului la familia de chip CNN utilizat [21,68]. Totodată, la etapele prezentate anterior, apare în plus şi etapa de achiziție a imaginii curente. Această etapă de achiziție nu modifică în esență nici algoritmul şi nici timpul total de procesare. Față de imaginile lipsite de zgomot care s-au utilizat la testarea prin simulare a algoritmului, în imaginile naturale apare zgomotul inerent din mediul real.

3.3.1 Selectarea şi urmărirea unui obiect într-o imagine binară

Se presupune cazul unei imagini binare de intrare în care se desemnează un obiect la un moment dat t_0, după un criteriu oarecare. În continuare, secvențele de imagini care urmează trebuie astfel prelucrate încât în imaginea rezultată să rămână numai obiectul desemnat anterior, dar în poziția curentă. Se va presupune, de asemenea, că obiectele din imaginile achiziționate, sunt separabile. Dacă obiectele sunt suprapuse, chiar şi parțial, sau sunt greu separabile, rețeaua le va trata ca fiind un singur obiect. În cele ce urmează se consideră că imaginile de intrare satisfac condițiile inițiale impuse de această etapă, fiind anterior preprocesate adecvat în acest scop.

Pentru desemnarea, selectarea şi urmărirea unui obiect într-o imagine binară se creează o imagine de "memorare" care conţine informaţia cu privire la obiectul desemnat sau selectat în eşantionul anterior. Această imagine de memorare la momentul iniţial t_0, pentru criteriul de desemnare ales, este o imagine cu cel puţin un pixel activ (+1) care se suprapune ca poziţie cu cel puţin un pixel activ al obiectului desemnat din planul de imagine achiziţionat. Începând de la momentul (t_0+T), imaginea de memorare va fi imaginea de ieşire de la secvenţa anterioară [t_0+(k-1)T]. Se continuă astfel cu secvenţele de imagini de intrare care se achiziţionează cu pasul de eşantionare T.

Pentru selectarea automată a unui obiect desemnat dintr-o imagine binară prin procesarea cu un sigur operator, s-a folosit operatorul follow.tem. Dimensionarea acestuia s-a făcut euristic de către autor. Pentru stratul 1 de reţea, imaginea de intrare curentă se aplică pe INPUT, iar imaginea de memorare se aplică pe STATE.

Operatorul follow.tem este definit de relaţiile: (3.4)

$$A = \begin{bmatrix} 0.3 & 0.3 & 0.3 \\ 0.3 & 4 & 0.3 \\ 0.3 & 0.3 & 0.3 \end{bmatrix} \quad B = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 5.1 & 0 \\ 0 & 0 & 0 \end{bmatrix} \quad z = \boxed{0}$$

Pentru o succesiune de imagini de intrare şi imagini de memorare ca în figura 3.10, se prezintă în aceiaşi figură imaginile de ieşire după procesarea cu operatorul follow.tem.

Se constată ca răspunsul reţelei este corect, în imaginea de ieşire fiind prezent numai obiectul desemnat la început şi poziţionat corespunzător imaginii curente. Diferenţa între două poziţii consecutive ale obiectului marcat poate fi de mai mulţi pixeli pe fiecare direcţie.

Procesarea cu follow.tem este corectă şi în cazul desemnării unui obiect de dimensiuni mai mici, de câţiva pixeli, indiferent în ce poziţie s-ar găsi, indiferent ce mişcare execută, de rotaţie, de translaţie sau şi una şi alta (figura 3.11; 1a-1c şi 2a-2c).

Pentru urmărirea corectă a unui obiect într-o secvenţă de imagini binare, condiţia necesară şi suficientă este ca să existe cel puţin un element activ comun al obiectului din imagini succesiv achiziţionate. În caz contrar răspunsul reţelei este greşit, adică se pierde obiectul din imaginea de ieşire.

Odată ce obiectul a fost pierdut, nu va mai exista obiect selectat nici în imaginea de memorare pentru eşantionul următor. Urmărirea acelui obiect va fi posibilă numai prin reluarea procedurii de desemnare de la momentul iniţial t_0.

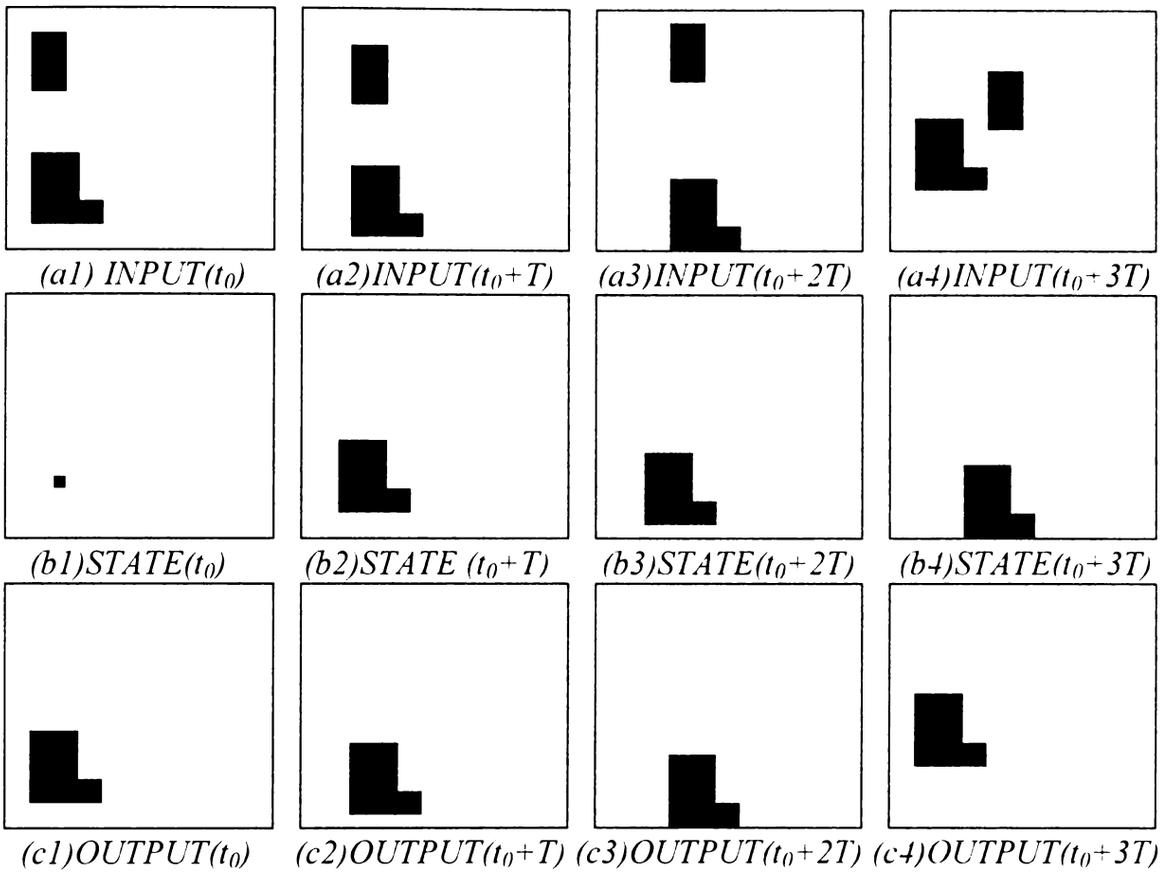


Figura 3.10. Procesarea cu follow.tem a unui şir de imagini la (t_0+kT),k=0,1,2,3: (a1-a4) imagini de intrare binare preprocesate; (b1-b4) imagini de memorare(t_0+kT)=ieşire [t_0+(k-1)T]; (c1-c4) imagini de ieşire.

În cazul în care a fost desemnat un obiect şi dintr-o cauză oarecare acesta ajunge ca urmare a mişcării sale să se suprapună în imagine cu un alt obiect, se constată că reţeaua va urmări, până este posibil, ambele obiecte, inclusiv şi cel care nu a fost desemnat (figura 3.11; 3a-3c şi 4a-4c). Şi în acest caz rezultă necesitatea unei noi desemnări a obiectului dorit.

Dacă pasul de eşantionare T este exprimat în secunde, la dimensiunea obiectului desemnat de N*N pixeli, viteza maximă pe o direcţie a obiectului, acceptată de către operatorul follow.tem pentru procesare corectă, este (N/T) pixeli/ secundă.

În cazul utilizării operatorului follow.tem timpul de procesare este deci dependent de caracterul imaginii de intrare, de dimensiunea obiectului care s-a selectat şi de diferenţa între poziţiile obiectului desemnat la două imagini succesiv achiziţionate. Timpul de procesare este proporţional cu numărul de iteraţii necesar. Pentru un obiect cu dimensiunea N*N pixeli rezultă cel mult N^2 iteraţii.

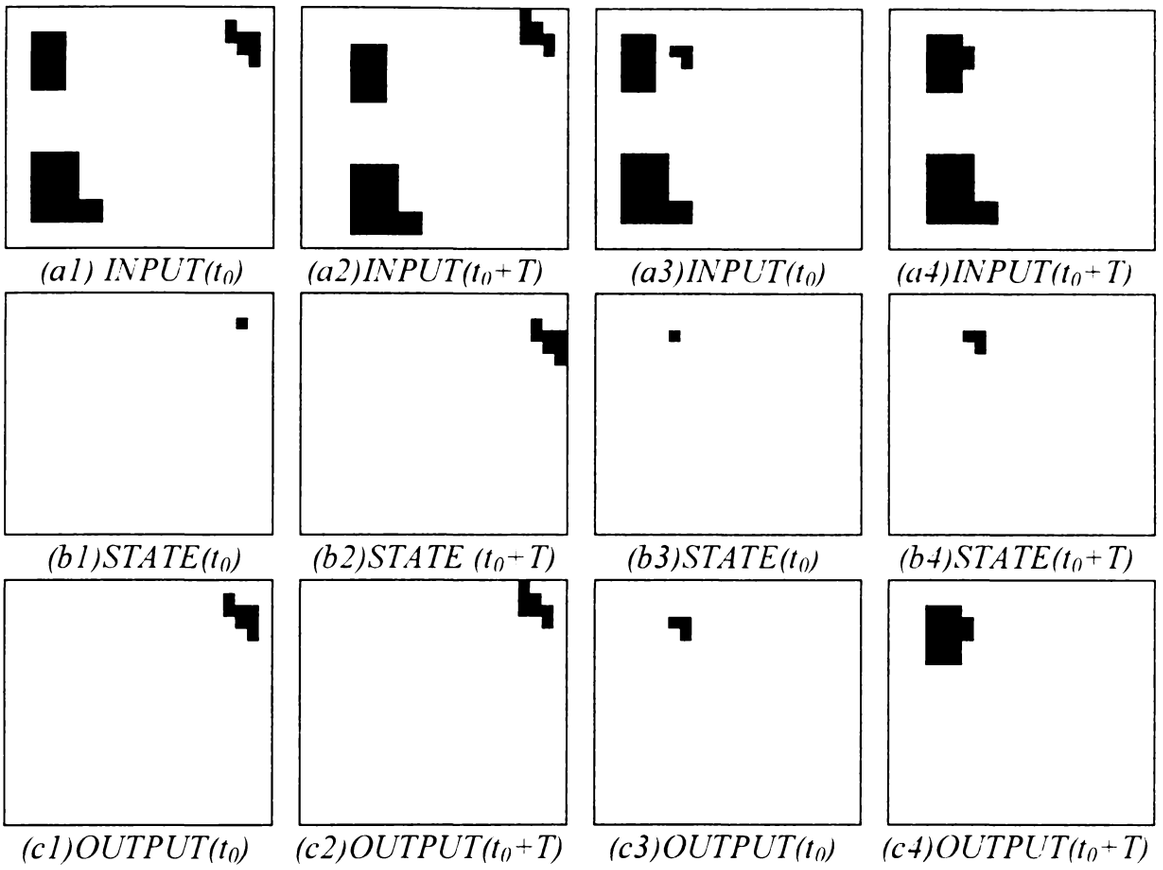


Figura 3.11. Procesarea cu follow.tem a unui şir de imagini la (t_0+kT), k=0,1: (a1-a4) imagini de intrare binare preprocesate; (b1-b4) imagini de memorare(t_0+kT)= imagini de ieşire [t_0+(k-1)T]; (c1-c4) imagini de ieşire.

La implementarea pe un chip-CNN [21], în cazul optimizării operatorului follow.tem se impune în primul rând condiția ca în imaginea de ieşire rezultată să fie numai obiectul desemnat, unul singur. Se acceptă ca reproducerea formei după procesare să nu fie chiar corectă la o anumită secvență achiziționată, dacă deplasarea după o anumită direcție se face cu mai mult decât un pixel într-o perioadă de eşantionare, dar la secvența următoare să se corecteze şi forma obiectului. Astfel, în urma optimizării operatorului follow.tem la implementarea pe un chip-CNN de 20*22 pixeli, rezultă operatorul follow1.tem definit de relațiile: (3.5)

$$A = \begin{bmatrix} 0.275 & 0.275 & 0.275 \\ 0.275 & 2.3 & 0.275 \\ 0.275 & 0.275 & 0.275 \end{bmatrix} \quad B = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 3 & 0 \\ 0 & 0 & 0 \end{bmatrix} \quad z = \boxed{0}$$

Din figura 3.12 rezultă că urmărirea este corectă la fiecare pas de eşantionare, rezultând întotdeauna un singur obiect, cel care a fost desemnat.

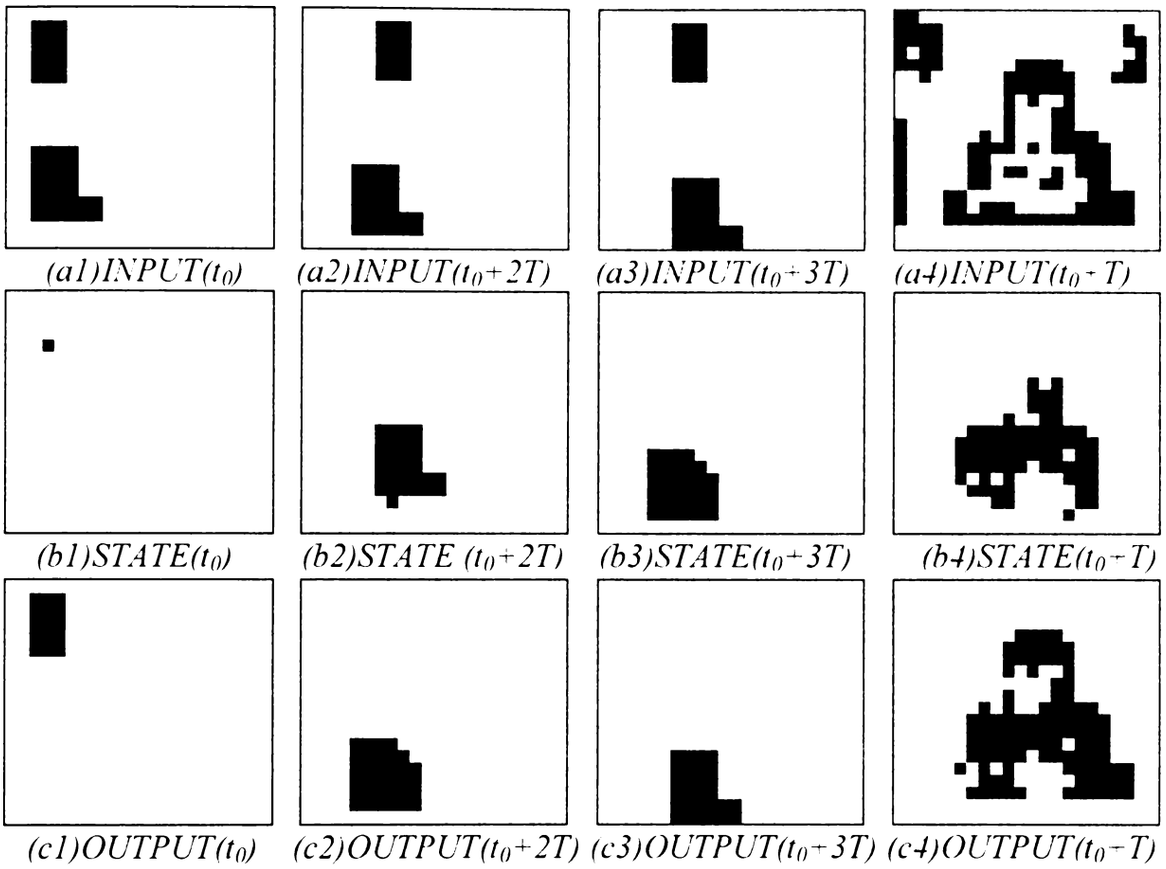


Figura 3.12. Procesarea cu follow1.tem a unor imagini la (t_0+kT),k=0,1,2: (a1-a4) imagini de intrare binare preprocesate; (b1-b4) imagini de memorare(t_0+kT)= imagini de ieşire [t_0+(k-1)T]; (c1-c4) imagini de ieşire.

Reproducerea formei obiectului marcat este corectă dacă deplasarea obiectului pe o direcţie nu a fost mai mare de un pixel, în intervalul de timp dintre două imagini consecutive de intrare. Pentru deplasări mai mari erorile de reproducere a formei cresc, dar urmărirea rămâne corectă. Forma obiectului se va corecta după secvenţele de imagini achiziţionate care vor urma. Obiecte de dimensiuni mici sunt urmărite corect, dar forma acestora se reproduce cu erori relative mai mari raportate la dimensiunile obiectului, respectiv are loc mai uşor pierderea de sub urmărire.

La urmărirea unui obiect, pentru a nu creşte considerabil timpul de procesare corespunzător primului pas de eşantionare, se acceptă introducerea unei erori de reproducere de formă. Această eroare se poate micşora prin mărirea în imaginea de memorare iniţială, a numărului de elemente active la mai mulţi pixeli.

În general, la implementarea pe un chip-CNN, pentru funcţionarea corectă a urmăririi şi reproducerii formei unui obiect selectat, se impun condiţii restrictive mai severe privind valorile maxime ale dimensiunii orificiilor din obiect, distanţa dintre obiecte, viteza de deplasare (figura 3.12; 4a-4c).

Pe baza rezultatelor obținute la selectarea unui obiect dintr-o imagine prin procesarea unor imagini cu operatorul follow1.tem pe un chip-CNN [21], având în vedere dimensiunile obiectelor în raport cu dimensiunile imaginii, se poate aprecia totuşi că erorile relative introduse de către acest operator sunt acceptabile.

3.3.2 Determinarea poziției unui obiect în planul imaginii utilizând rețele neuronale celulare

Având o imagine binară de intrare v(i,j) de dimensiune M*N, definită prin relația (3.1), care conține un singur obiect, se doreşte să se determine poziția obiectului în planul imaginii.

Se va considera că poziția obiectului în planul imaginii se identifică cu poziția punctului central al obiectului. Prin determinarea poziției P(x,y,kT) a obiectului în planul imaginii se va înțelege numai identificarea punctului central fără a cunoaşte numeric coordonatele acestuia în sistemul de referință rectangular xOy ataşat planului de imagine.

Pentru determinarea poziției punctului central al obiectului se va folosi familia de operatori "center.tem"[125]. Prin utilizarea familiei de operatori center.tem se realizează de fapt o decojire simetrică succesivă a obiectului cu câte un pixel după fiecare direcție (V, N-V, N, N-E, E, S-E, S, S-V). Acest proces continuă până când obiectul va fi format dintr-un singur pixel activ (+1). Prin identificarea poziției acestui pixel se consideră determinată şi poziția punctului central al obiectului din planul imaginii achiziționate. Imaginea de intrare binară este adusă pe INPUT, iar imaginea STATE este indiferentă (pixelii pot avea orice valori în domeniul standard CNN). Familia de operatori center.tem este alcătuită din operatorii center1.tem-center8.tem definiți de relațiile [125]: (3.6)

center1.tem

$$A = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{bmatrix} \quad B = \begin{bmatrix} 1 & 0 & 0 \\ 1 & 4 & -1 \\ 1 & 0 & 0 \end{bmatrix} \quad z = \boxed{-1}$$

center2.tem

$$A = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{bmatrix} \quad B = \begin{bmatrix} 1 & 1 & 1 \\ 1 & 6 & 0 \\ 1 & 0 & -1 \end{bmatrix} \quad z = \boxed{-1}$$

center3.tem

A =				B =			z =
	0	0	0	1	1	1	-1
	0	1	0	0	4	0	
	0	0	0	0	-1	0	

center4.tem

A =				B =			z =
	0	0	0	1	1	1	-1
	0	1	0	0	6	1	
	0	0	0	-1	0	1	

center5.tem

A =				B =			z =
	0	0	0	0	0	1	-1
	0	1	0	-1	4	1	
	0	0	0	0	0	1	

center6.tem

A =				B =			z =
	0	0	0	-1	0	1	-1
	0	1	0	0	6	1	
	0	0	0	1	1	1	

center7.tem

A =				B =			z =
	0	0	0	0	-1	0	-1
	0	1	0	0	4	0	
	0	0	0	1	1	1	

center8.tem

A =				B =			z =
	0	0	0	1	0	-1	-1
	0	1	0	1	6	0	
	0	0	0	1	1	1	

Pentru a se ajunge la sfârşitul prelucrării cu familia de operatori center.tem doar la un singur pixel activ, numărul de iteraţii pentru decojire trebuie corelat cu dimensiunea obiectului. Astfel, la obiectele cu dimensiuni mai mari, pentru obţinerea unui rezultat corect numărul de iteraţii trebuie crescut considerabil. Această familie de operatori este implementabilă pe chipul-CNN existent [21], dar chiar şi în această situaţie procesarea este foarte mare consumatoare de timp.

În figura 3.13 se prezintă determinarea poziţiei punctului central al obiectului dintr-o imagine de intrare binară de 20*22 pixeli, după 25 de iteraţii respectiv rezultatul final, adică imaginea de ieşire după 50 de iteraţii.

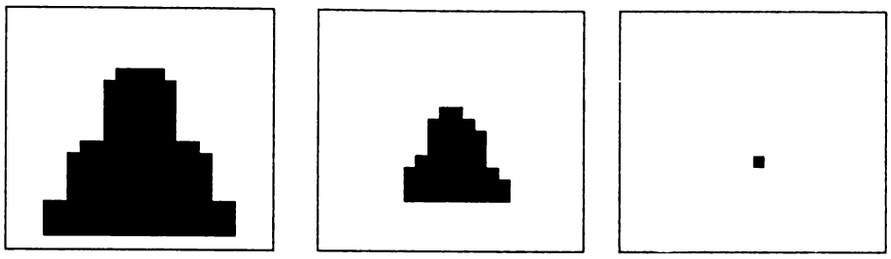


Figura 3.13. Determinarea punctului central cu central .tem pentru un obiect fără orificii: (a) imaginea de intrare;(b) imaginea de ieşire după 25 de iteraţii; (c) imaginea de ieşire după 50 de iteraţii.

Pentru obţinerea unui rezultat convenabil cu operatorii center.tem se presupune, de asemenea, că obiectul care este decojit are un contur închis format numai din elemente active, care au valoarea +1. Dar dacă obiectul desemnat nu este cu contur închis şi are şi orificii sau goluri, de exemplu datorită iluminării necorespunzătoare din imaginea reală achiziţionată, metoda determinării poziţiei punctului central prin acest procedeu nu conduce la un rezultat corect. În figura 3.14 se prezintă o astfel de situaţie, când nu rezultă univoc poziţia punctului central, adică un singur pixel şi astfel nu se poate preciza exact poziţia în plan a obiectului. Această eroare este inacceptabilă pentru funcţionarea corectă a algoritmului de urmărire a unui obiect aflat în mişcare.

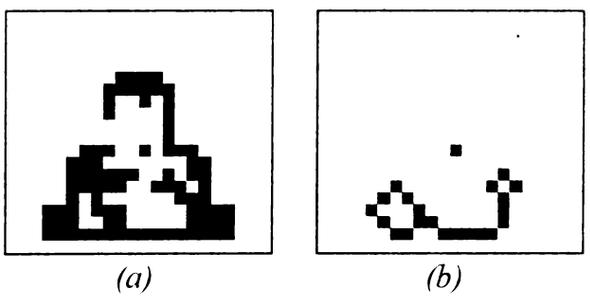


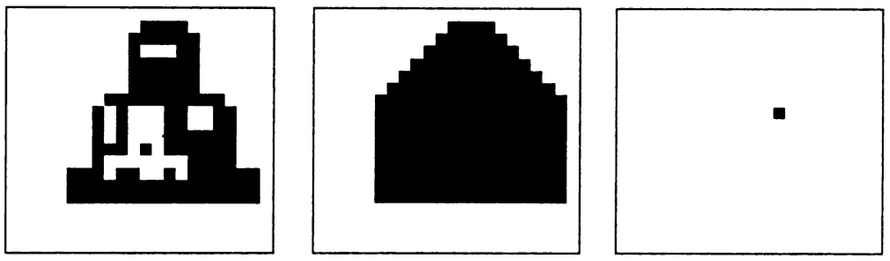
Figura 3.14. Determinarea punctului central cu central .tem pentru un obiect cu orificii: (a) imaginea de intrare; (b) imaginea de ieşire după 50 de iteraţii.

În consecinţă, înaintea determinării poziţiei punctului central cu ajutorul familiei de operatori center.tem, este necesară realizarea unei preprocesări cu ajutorul operatorului hollow.tem [125] a imaginii de intrare binare, pentru obţinerea obiectului cu contur închis şi fără orificii. Imaginea binară conţinând un singur obiect se aplică pe INPUT, iar pe STATE se aplică "imaginea +1", adică toate elementele de imagine sunt de valoarea +1.

Operatorul hollow.tem este definit de relațiile [125]: (3.7)

$$A = \begin{bmatrix} 0.5 & 0.5 & 0.5 \\ 0.5 & 2 & 0.5 \\ 0.5 & 0.5 & 0.5 \end{bmatrix} \quad B = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 2 & 0 \\ 0 & 0 & 0 \end{bmatrix} \quad z = \boxed{3.25}$$

În figura 3.15 se prezintă un exemplu de determinare a poziției punctului central al unui obiect cu orificii, după procesare cu hollow.tem, rezultând în acest caz un singur pixel, adică o determinare corectă a poziției punctului central al obiectului.



(a) *(b)* *(c)*

Figura 3.15. Determinarea punctului central pentru un obiect cu orificii: (a) imaginea de intrare; (b) imaginea de ieşire după hollow.tem; (b) imaginea de ieşire după center.tem.

Prin utilizarea familiei de operatori center.tem nu rezultă însă o soluție unică, poziția punctului final activ depinzând de direcția din care se începe decojirea simetrică. Se poate constata că datorită acestui motiv rețeaua poate greşi cu un pixel poziționarea (de exemplu, pe verticală, figura 3.15).

Numărul de iterații necesare pentru o procesare corectă cu hollow.tem este dependent de dimensiunea obiectului selectat şi de numărul de orificii care există în obiect. În consecință, datorită preprocesării cu ajutorul hollow.tem a imaginii binare de intrare pentru a putea determina corect poziția punctului central al unui obiect cu center.tem este necesară şi mai mult timp.

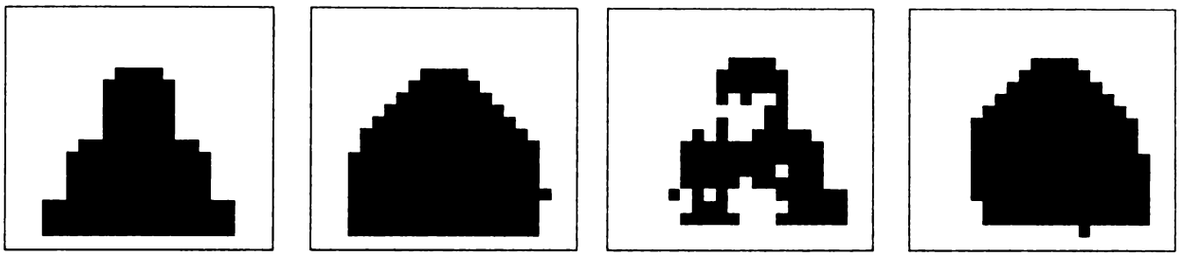
La implementarea pe un chip-CNN de 20*22 de pixeli a operatorului hollow.tem, se doreşte, de asemenea, ca răspunsul obținut să fie cât mai apropiat de răspunsul ideal obținut prin simulare, pentru a nu modifica substanțial poziția punctului central. De aceea, după optimizarea operatorului hollow.tem rezultă operatorul hollow1.tem definit de relațiile: (3.8)

A = | 0.5 | 0.5 | 0.5 |
| --- | --- | --- |
| 0.5 | 2.0 | 0.5 |
| 0.5 | 0.5 | 0.5 |

B = | 0 | 0 | 0 |
| --- | --- | --- |
| 0 | 2 | 0 |
| 0 | 0 | 0 |

z = 2.750

În figura 3.16 sunt prezentate rezultatele obținute la testarea pe un chip-CNN de 20*22 de pixeli a operatorului hollow1.tem.



(a) (b) (c) (d)

*Figura 3.16. Testarea pe chip-CNN de 20*22 de pixeli a hollow1.tem:*
(a) şi (c) imagini de intrare; (b) şi (d) imagini de ieşire după procesare.

3.3.3 Determinarea unghiurilor de referință θ1 şi θ2

Având în planul imaginii stratului 1 poziția punctului central al unui obiect se pot determina în etapa a 3-a θ1 şi θ2, care reprezintă valorile unghiurilor de referință pentru poziția următoare a cuplelor cinematice de rotație A şi B ale brațului robotic, asemănător soluțiilor pentru problema cinematică inversă. Se creează în acest scop imaginile Δθ1 şi Δθ2, adică straturile 2 şi 3 ale structurii rețelei asociate algoritmului de urmărire a unui obiect. Unghiurile de referință noi, θ1 şi θ2, rezultă pe baza relațiilor (3.2) şi (3.3).

Analizând transformata cinematică inversă pentru brațul unui robot cu cuple cinematice de rotație cu două grade de libertate [81], imaginile unghiurilor Δθ1 respectiv Δθ2, propuse de autor, pot avea una din următoarele forme sau combinații ale acestora:

i.) Imagini cu niveluri de gri (figura 3.17:a şi b), în care fiecare pixel are o valoare diferită şi prin nivelul său de gri conține informația referitoare la mărimile Δθ1 şi Δθ2. În imaginile Δθ1 şi Δθ2 valorile pixelilor depind de coordonatele x şi y ale elementului de imagine în sistemul de referință ataşat planului de imagine Δθ1=f(x,y) şi Δθ2=g(x,y). În acest caz nu se introduc aproximări dar este nevoie de un timp mai lung de procesare pentru determinarea unghiurilor de referință θ1 respectiv θ2. Timpul de

procesare este mai mare mai ales datorită necesității obținerii punctului central al obiectului selectat.

ii.) Imagini cu niveluri de gri, care conțin pixeli cu valori identice pe câte o coloană (pentru imaginea Δθ1, figura 3.17c) respectiv câte o linie (pentru imaginea Δθ2, figura 3.17d). Fiecare coloană respectiv linie, prin nivelul său de gri conține informația referitoare la mărimile Δθ1 şi Δθ2, dacă sunt valabile aproximările Δθ1=f(x), adică depinde numai de poziția după axa x şi Δθ2=g(y), adică depinde numai de poziția după axa y. Aceste aproximări permit ca imaginile să devină chiar binare, astfel încât fiecare coloană (pentru imaginea Δθ1, figura 3.17e) respectiv fiecare linie (pentru imaginea Δθ2, figura 3.17f) poate conține informația corespunzătoare codificată binar.

iii.) Imagini binare, pentru cea mai simplă situație, când imaginile unghiurilor Δθ1 (figura 3.17g) respectiv Δθ2 (figura 3.17h) servesc doar la determinarea semnului de variație ale unghiurilor față de poziția curentă, cu o mărime prestabilită, constantă. Prin această aproximare se poate chiar elimina din algoritm necesitatea de a determina punctul central al obiectului.

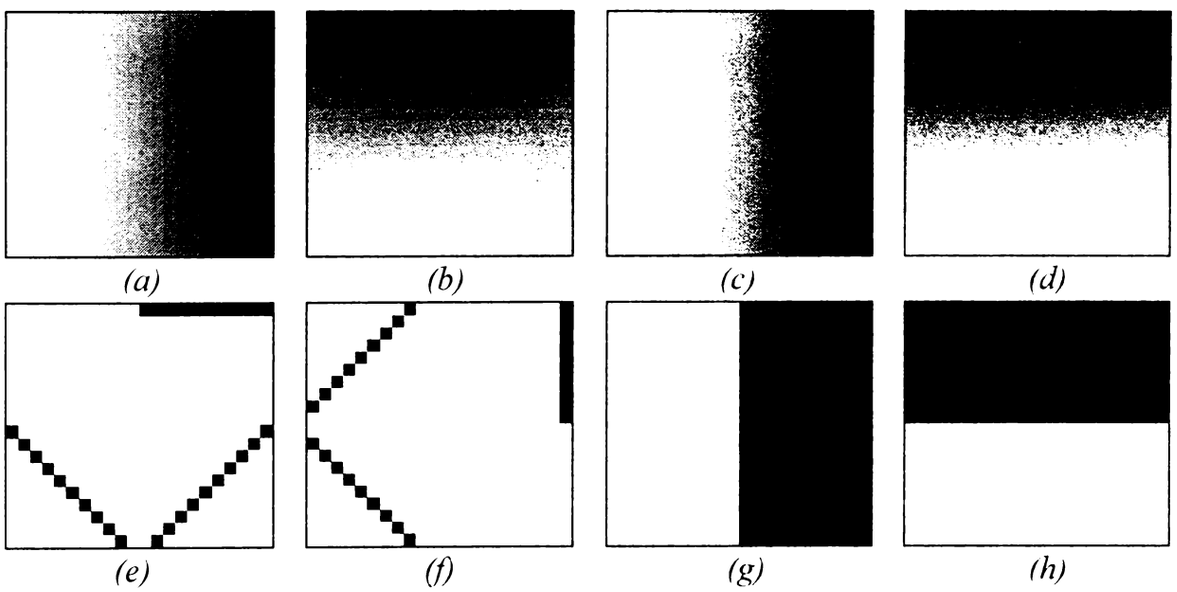


Figura 3.17. Imaginile unghiurilor de comandă: (a) imaginea Δθ1, cu niveluri de gri; (b) imaginea Δθ2, cu niveluri de gri; (c) imaginea Δθ1, cu niveluri de gri având coloane identice; (d) imaginea Δθ2, cu niveluri de gri având linii identice; (e) imaginea binară Δθ1; (f) imaginea binară Δθ2; (g) şi(h) imagini utilizate la determinarea semnului Δθ1 şi Δθ2.

La alegerea structurii imaginilor Δθ1 şi Δθ2 se au în vedere următoarele aspecte:

- precizia şi viteza impusă pentru poziționarea punctului central al obiectului față de originea sistemului de referință ataşat planului de imagine;
- viteza de deplasare a obiectului pe fiecare axă;
- modul în care se realizează implementarea pe chip-CNN a algoritmului de urmărire al unui obiect aflat în mişcare.

Prin simulare soft sau la procesarea pe un chip-CNN 64*64 de pixeli [68] se pot folosi oricare din variantele de imagini $\Delta\theta 1$ şi $\Delta\theta 2$ prezentate anterior, pe când la procesarea pe un chip CNN 20*22 de pixeli, pot fi utilizate doar imaginile binare [21].

Indiferent de structura imaginilor cu care se operează, procedeul de determinare a mărimilor $\Delta\theta 1$ şi $\Delta\theta 2$ se bazează pe extragerea, cu metode CNN, a valorilor corespunzătoare unui element de imagine, sau corespunzătoare unei linii ori coloane din imaginile asociate straturilor 2 şi 3, pozițiile acestora fiind identificate prin pixeli de valoare +1 în planul imaginii stratului 1.

Valorile elementelor din imaginile $\Delta\theta 1$ şi $\Delta\theta 2$, cu niveluri de gri, trebuie determinate pixel cu pixel, corespunzător pozițiilor pe care le poate lua brațul robotic. Imaginile $\Delta\theta 1$ şi $\Delta\theta 2$ se pot realiza şi în cazul în care se cunosc valorile elementelor de imagine doar pentru un număr redus de pixeli. Valorile elementelor de imagine necunoscute pot fi determinate prin interpolare, de exemplu, prin interpolare spline cubică utilizând rețele neuronale celulare, metodă propusă de autor în capitolul 4.

Pentru cazul în care la imaginile unghiurilor $\Delta\theta 1$ şi $\Delta\theta 2$ se folosesc imagini imagine cu niveluri de gri (figura 3.17;a şi b), valorile acestor unghiuri se determină pe baza posibilității utilizării imaginii binare mască (MASK), la procesarea unei imagini cu ajutorul unui operator. Dacă într-o imagine mască binară un singur element este inactiv (de exemplu, cu valoarea +1, figura 3.18a), dintr-o altă imagine cu niveluri de gri (figura 3.18b), care coincide ca dimensiuni cu prima, se poate extrage valoarea unui pixel care se află în aceiaşi poziție în imaginea a doua ca şi elementul inactiv din prima imagine binară. Pentru determinarea valorii unui pixel dintr-o imagine cu niveluri de gri se va utiliza operatorului filwhite.tem sau operatorul filblack.tem [125].

Operatorul filwhite.tem este definit de relațiile: (3.9)

$$A = \begin{array}{|c|c|c|}\hline 0 & 0 & 0 \\ \hline 0 & 2 & 0 \\ \hline 0 & 0 & 0 \\ \hline \end{array} \qquad B = \begin{array}{|c|c|c|}\hline 0 & 0 & 0 \\ \hline 0 & 0 & 0 \\ \hline 0 & 0 & 0 \\ \hline \end{array} \qquad z = \boxed{-4}$$

Operatorul filblack.tem este definit de relațiile: (3.10)

$$A = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 2 & 0 \\ 0 & 0 & 0 \end{bmatrix} \qquad B = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix} \qquad z = \boxed{4}$$

În urma prelucrării, fără imagine mască, a unei imagini cu operatorul filwhite.tem sau cu operatorul filblack.tem, vor rezulta imagini care au toți pixelii de valoare -1 respectiv +1. Utilizând operatorul filwhite.tem sau filblak.tem, dacă se folosește și imagine mască binară, în imaginea cu niveluri de gri, care rezultă în urma prelucrării, un singur element va diferi de valoarea -1 respectiv +1 (figura 3.18;c și d). Doar acestui element din imagine i s-a menținut valoarea inițială dinaintea prelucrării, prin elementul inactiv situat în aceiași poziție din imaginea mască. Astfel, printr-o singură instrucțiune AMC, se poate extrage valoarea acestui element ca fiind valoarea maximă respectiv minimă din imagine (figura 3.18). Ca și în cazul imaginilor prezentate în figura 3.17;a;b, se parcurg practic aceleași etape pentru cunoașterea valorii unei coloane sau a unei linii în imaginile prezentate în figura 3.17;c;d, imaginile mască vor avea în aceste cazuri o coloană respectiv o linie inactivă.

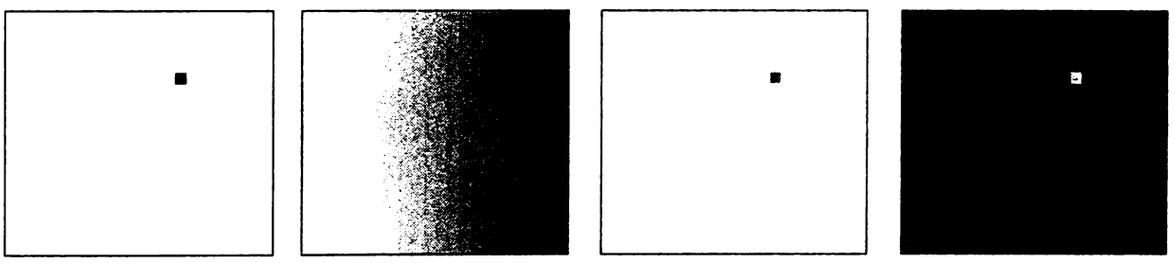
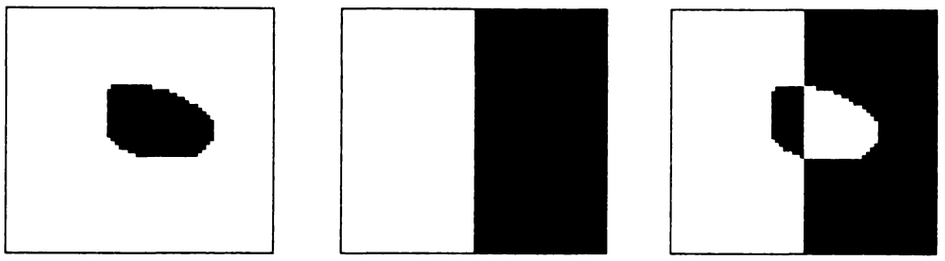


Figura 3.18. Determinarea unghiului Δθ1 pentru imagini cu niveluri de gri: (a) imaginea binară mască (stratul 1); (b) imaginea Δθ1, cu niveluri de gri; (c) imaginea de ieșire după filwhite.tem; (d) imaginea de ieșire după filblack.tem.

În cazul utilizării imaginilor din figura 3.17;g;h, pentru determinarea semnelor unghiurilor Δθ1 și Δθ2, se utilizează proprietatea acestora conform căreia valoarea medie a pixelilor este nulă, adică numărul de pixeli albi (-1) este identic cu numărul de pixeli negri (+1). Printr-o singură instrucțiune AMC se poate contoriza numărul de pixeli de un anumit fel din imaginea binară (de valori +1 sau valori -1). Utilizând numai procesări elementare logice (XOR) între imaginea binară cu un singur obiect, stratul 1 și imaginile unghiurilor Δθ1 și Δθ2, se poate pune în evidență modificarea numărului de pixeli de un anumit fel, dacă obiectul nu este situat în centrul imaginii. Semnele pentru

$\Delta\theta 1$ şi $\Delta\theta 2$ rezultă prin compararea numărului de elemente de un anumit fel cu valoarea medie din imaginea originală.

În figura 3.19 se prezintă un exemplu pentru determinarea semnului pentru $\Delta\theta 1$. Dacă pragul de comparaţie este diferit de zero şi diferit pentru sensul pozitiv respectiv sensul negativ, se poate realiza o histereză pentru comanda unghiului cuplei, adică camera video să se mişte numai la modificarea substanţială a poziţiei obiectului urmărit în raport cu centrul imaginii.



(a) *(b)* *(c)*

Figura 3.19. Determinarea semnului $\Delta\theta 1$: (a) imaginea binară de pe stratul 1; (b) imaginea binară $\Delta\theta 1$; (c) imaginea de ieşire după XOR.

Având în vedere caracteristicile circuitului CNN cu 20*22 de pixeli [21], pentru a putea implementa pe acest chip, algoritmul de urmărire a unui obiect aflat în mişcare, imaginile alese pentru determinarea unghiurilor $\Delta\theta 1$ şi $\Delta\theta 2$ sunt cele din figura 3.17:e;f. Astfel, dacă ne referim la imaginea $\Delta\theta 1$, fiecărei coloane îi corespunde o anumită valoare pentru poziţionarea pe orizontală, cuprinsă între o valoare maximă şi o valoare minimă. Partea dreaptă a imaginii corespunde valorilor pozitive ale unghiului $\Delta\theta 1$, în stânga imaginii sunt codurile valorilor negative ale unghiului.

Pentru imaginea $\Delta\theta 2$ fiecare linie reprezintă câte o valoare necesară pentru poziţionarea pe verticală, între o valoarea minimă şi valoarea maximă. În partea de sus a imaginii sunt valorile pozitive ale unghiului $\Delta\theta 2$, iar în partea de jos valorile negative ale aceluiaşi unghi. Numărul de poziţii discrete este diferit în cele două imagini, deoarece dimensiunile imaginii nu sunt identice după cele două direcţii (având dimensiunea de 20*22 pixeli). Faţă de $\Delta\theta 2$, apar în plus două poziţii discrete la $\Delta\theta 1$.

Astfel, pornind de la poziţia punctului central reprezentată în stratul 1 al reţelei, pentru determinarea noilor valori pentru $\Delta\theta 1$ şi $\Delta\theta 2$ se parcurg mai mulţi paşi de procesare, ca în figura 3.20. Pentru extragerea valorilor $\Delta\theta 1$ şi $\Delta\theta 2$ se folosesc operatorii shsiud.tem sau shsirl.tem care s-au determinat pe baza shadsim.tem [125]. Dacă o imagine binară (aplicată pe STATE) este procesată cu ajutorul operatorilor

shsiud.tem sau shsirl.tem, în imaginea rezultată, dimensiunea obiectelor (de valori +1) se va mări cu câte un pixel prin crearea unor umbre pe direcția N-S respectiv pe direcția E-V.

Operatorul shsiud.tem este definit de relațiile [125]: (3.11)

$$A = \begin{bmatrix} 0 & 1 & 0 \\ 0 & 2 & 0 \\ 0 & 1 & 0 \end{bmatrix} \quad B = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix} \quad z = \boxed{2}$$

Operatorul shsirl.tem este definit de relațiile [125]: (3.12)

$$A = \begin{bmatrix} 0 & 0 & 0 \\ 1 & 2 & 1 \\ 0 & 0 & 0 \end{bmatrix} \quad B = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix} \quad z = \boxed{2}$$

Prin prelucrarea cu operatorul shsiud.tem a unei imagini binare cu un singur obiect (figura 3.20d) de dimensiunea unui pixel (punctul central), se obține o imagine care conține un obiect de dimensiunea unei coloane (figura 3.20e), iar prin procesarea cu operatorul shsirl.tem a aceleaşi imagini inițiale rezultă o imagine care conține un obiect de dimensiunea unei linii (figura 3.20i).

Pentru determinarea lui $\Delta\theta 1$ se selectează coloana curentă (figura 3.20g), din imaginea $\Delta\theta 1$ prin realizarea funcției logice ŞI, pixel cu pixel, între imaginea care conține un obiect de dimensiunea unei coloane şi imaginea $\Delta\theta 1$. Cele două imagini se aplică pe INPUT respectiv pe STATE . Funcția logică ŞI se realizează cu operatorul logAND.tem definit de relațiile[125]: (3.13)

$$A = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 2 & 0 \\ 0 & 0 & 0 \end{bmatrix} \quad B = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{bmatrix} \quad z = \boxed{-1}$$

Pentru ca indiferent de coloana selectată din imaginea $\Delta\theta 1$ să se obțină în acelaşi mod şi acelaşi loc codul unghiului $\Delta\theta 1$, asupra imaginii care rezultă după funcția logică ŞI, se mai aplică o nouă procesare de umbră E-V, cu operatorul shsirl.tem. Valoarea analogică pentru $\Delta\theta 1$ va rezulta întotdeauna prin decodificarea coloanei extreme din dreapta sau stânga imaginii rezultate (figura 3.20h).

În cazul determinării valorii $\Delta\theta 2$, procedeul de selectare este asemănător cu cel prezentat pentru unghiul $\Delta\theta 1$, dar se va selecta o linie din imaginea $\Delta\theta 2$ (figura 3.20k).

Valoarea analogică pentru Δθ2 rezultă prin decodificarea primei sau a ultimei linii de sus a imaginii care rezultă după efectuarea cu operatorul shsiud.tem a umbrei N-S asupra imaginii cu o linie selectată din Δθ2 (figura 3.20l).

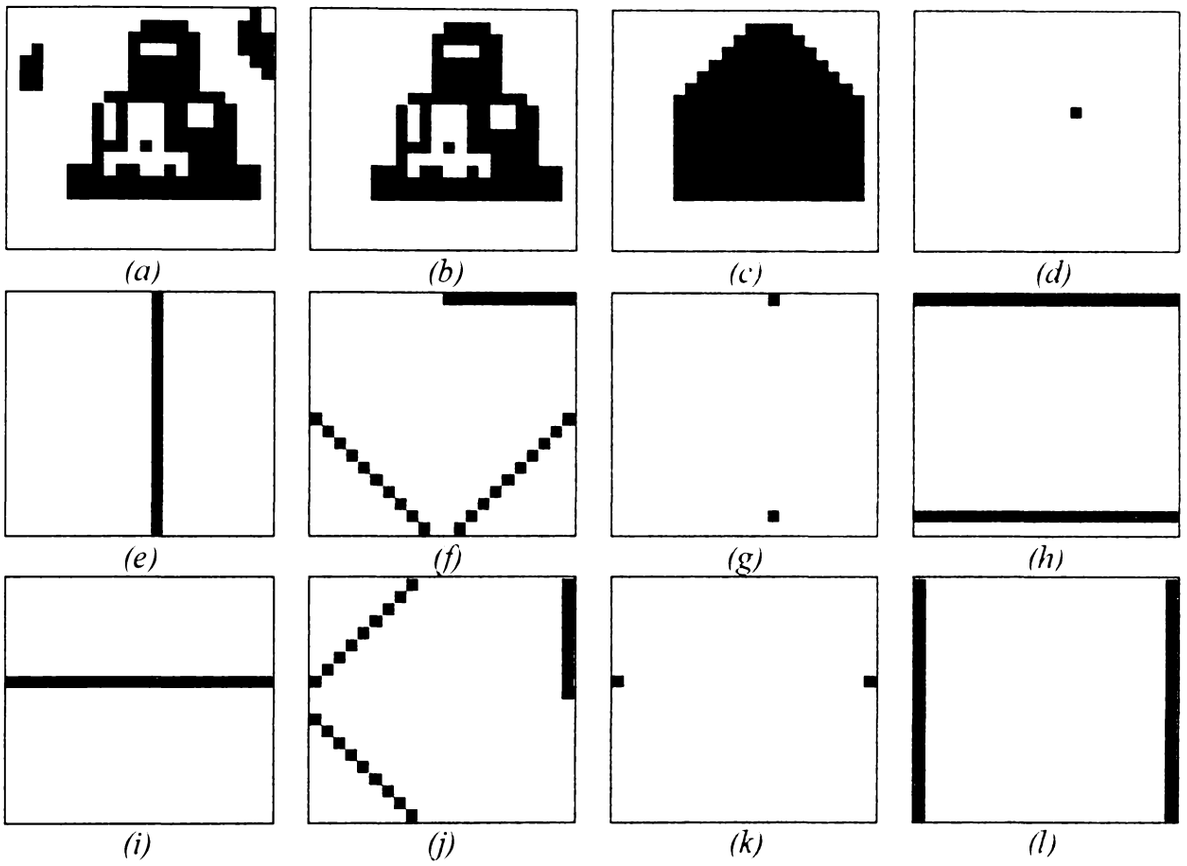


Figura 3.20. Simularea algoritmului CNN de urmărire al unui obiect aflat în mişcare: a) imaginea actuală de intrare; b) imaginea de ieşire după follow.tem; c) imaginea de ieşire după hollow.tem; d) imaginea de ieşire cu punctul central al obiectului; e) imaginea de ieşire după shsiud.tem; f) imaginea Δθl; g) imaginea de ieşire după logAND.tem; h) imaginea de ieşire după shsirl.tem; i) imaginea de ieşire după shsirl.tem; j) imaginea Δθ2; k) imaginea de ieşire după logAND.tem; l) imaginea de ieşire după shsiud.tem.

Deoarece procesările cu operatorii shsiud.tem şi shsirl.tem sunt tranziente, acestea necesită multe iteraţii, timpii de procesare depinzând de dimensiunile imaginii. Numărul de iteraţii pentru procesarea cu LogAND.tem este unu, deoarece aceasta este o procesare locală fără tranzient. În cazul unor imagini de dimensiunea 20*22 pixeli, pentru prelucrările cu operatorii shsiud.tem şi shsirl.tem numărul maxim de iteraţii necesar este de 22.

3.4 Testarea algoritmului CNN de urmărire a unui obiect aflat în mişcare

În figura 3.21 se prezintă sistemul experimental utilizat pentru testarea algoritmului de urmărire a unui obiect aflat în mişcare. Această structură de testare are la bază sistemul de dezvoltare "CadetWin"(CNN application development environment and toolkit under Windows) [123] şi camera video "EVI-D31 Sony" [132]. Camera video este mobilă, având două grade de libertate, asigurate de cuple cinematice de rotaţie. Comanda poziţiei se face cu ajutorul unui calculator printr-o interfaţa serială standard RS 232.

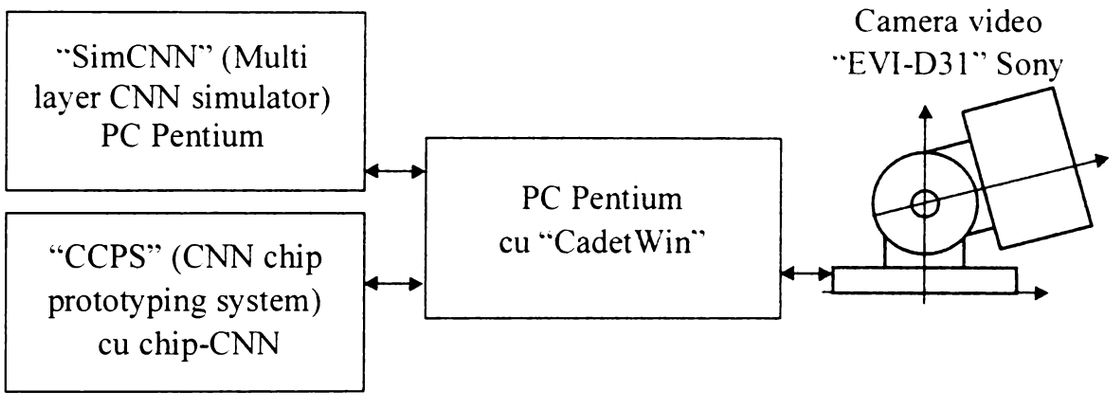


Figura 3.21. Sistemul experimental utilizat pentru testarea algoritmului de urmărire a unui obiect aflat în mişcare.

Algoritmul CNN de urmărire a unui obiect aflat în mişcare a fost implementat în program în limbaj de asamblare "AMC" (Extended analogic macro code and interpreter [130]) (anexa 1) şi testat integral cu ajutorul simulatorului "SimCNN"(Multi-layer CNN simulator with the visual mouse platform) [128]. Cu ajutorul acestei platforme, pe baza programelor AMC, este posibilă comanda cuplelor cinematice de rotaţie prin portul serial al camerei video prin valori absolute, valori relative sau cu valori prestabilite şi viteze reglabile. Pentru fiecare etapă din algoritmul analogic CNN pentru urmărirea obiectelor aflate în mişcare s-au utilizat la simulare şi soluţii care se pot implementa pe un chip-CNN cu 20*22 pixeli, cu intrare optică directă, existent la data respectivă [21].

Pentru implementarea pe chip-CNN a operatorilor follow1.tem şi hollow1.tem s-a utilizat interfaţa "CCPS"(CNN chip prototyping system) [131].

Principalele etape ale algoritmului CNN de urmărire a unui obiect aflat în mişcare sunt cele prezentate deja în figura 3.20.

În tabelul 3.1 se prezintă timpii de procesare caracteristici ai algoritmului CNN de urmărire a unui obiect aflat în mişcare, corespunzători simulării respectiv la implementarea pe un chip-CNN cu 20*22 de pixeli. S-a notat cu τ_{CNN} constanta de timp caracteristică pentru o celulă a acestui chip-CNN. În cazul simulării, când ecuația diferențială de stare care caracterizează dinamica unei celule este implementată numeric, timpul de procesare se va exprima în $\tau 1$; acesta reprezentă produsul dintre numărul de iterații necesare şi pasul de iterație.

Pentru operatorii center.tem, logAND.tem, shsirl.tem, shsiud.tem s-au utilizat aceleaşi valori ca şi cele care au fost stabilite la simulare, acestea fiind valabile chiar şi pentru situațiile cele mai defavorabile. Se observă că la operatorii follow1.tem şi hollow1.tem din etapa de implementare pe chip timpii de procesare sunt apropiați de valorile timpilor de procesare care au rezultat la simulare pentru follow.tem şi hollow.tem.

Operator	Sim CNN ($\tau 1$)	20*22 CNN-UM (τ_{CNN})
achiziție	-	~1
follow.tem	<50	40
center.tem	72	72
hollow.tem	<40	50
shsiud.tem	<22	<22
logAND.tem	1	1
shsirl.tem	<22	<22
shsirl.tem	<22	<22
logAND.tem	1	1
shsiud.tem	<22	<22
Total	<252	<253

*Tabel 3. 1:Timpii caracteristici de procesare ai algoritmului de urmărire a unui obiect aflat în mişcare la simulare şi la implementare pe chip-CNN cu 20*22 pixeli.*

Pentru sistemului robotic descris, în cazul circuitului CNN utilizat (cP400), constanta de timp care caracterizează o celulă fiind de τ_{CNN} = 250 ns, timpul total de procesare este de aproximativ 63.25 μs de la momentul achiziției imaginii până la obținerea unghiurilor de referință noi ale cuplelor cinematice.

Pe baza algoritmului propus de urmărire a unui obiect s-a realizat un program minimal de comandă, în limbaj de asamblare, care utilizează numai operații logice CNN ce se pot executa la nivel de CCPS, rezultând timpul cel mai scurt pentru execuția completă în buclă închisă a algoritmului de comandă (anexa 2). Cu sistemul prezentat în figura 3.21 s-a urmărit în acest fel mișcarea unui fascicol laser proiectat pe suprafața plană a unei fotografii. Programul, care cuprinde achiziția de imagini cu niveluri de gri de 300*300 pixeli, prelucrarea acestora conform algoritmului propus și furnizarea unghiurilor de comandă pentru camera video, se poate efectua în buclă cu o perioadă de 0.3s. În figura 3.22 se prezintă un cadru din imaginile achiziționate (record file) de la testarea în buclă a acestui program minimal AMC de urmărire a fascicolului laser cu ajutorul camerei video mobile.

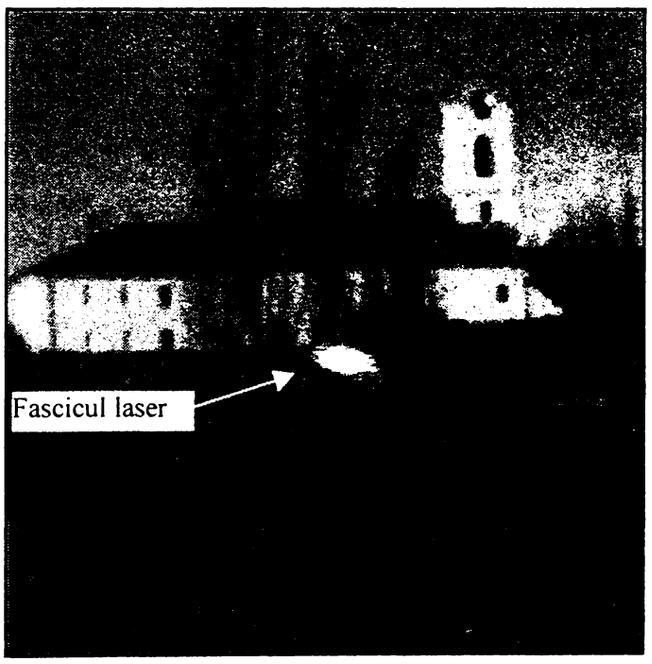


Figura 3.22. Cadru de imagine achiziționat la testarea algoritmului de urmărire a unui fascicol laser cu o cameră video.

Pe baza rezultatelor obținute prin simulare și respectiv implementare pe un chip-CNN a algoritmului de urmărire, rezultă că între două imagini de intrare succesiv achiziționate se pot efectua toate procesările necesare în vederea funcționării în timp real a sistemului robotic studiat. Creșterea performanțelor algoritmului CNN de urmărire a unui obiect aflat în mișcare, prin mărirea dimensiunilor imaginilor, va determina inevitabil și creșterea timpului de procesare dar poate fi realizabil în continuare în timp real având în vedere că majoritatea procesărilor efectuate sunt paralele.

3.5 Un nou algoritm pentru determinarea punctului central al unui obiect utilizând rețele neuronale celulare

Aşa cum s-a prezentat în paragraful 3.3.2, determinarea punctului central pentru un obiect dintr-o imagine reprezintă o etapă de procesare care se utilizează frecvent în algoritmi CNN de prelucrare complexă, în timp real [49,117]. Operatorii cunoscuți, center.tem şi hollow.tem, necesită însă un timp de procesare relativ ridicat. Cu toate acestea ei sunt utilizați, fiind în momentul de față singura procedură CNN care se poate aplica prin utilizarea numai de imagini binare.

În general, în cazul unor probleme complexe este dificil de proiectat un singur operator, liniar de dimensiune 3*3, care să ofere soluție printr-un singur pas de prelucrare, chiar dacă se pot procesa şi imagini cu niveluri de gri. De aceea, problema complexă se va descompune în probleme parțiale care deja se pot soluționa cu câte un singur operator elementar şi în timp foarte scurt, rezultând un algoritm CNN.

În cele ce urmează se propune un nou algoritm pentru determinarea punctului central al unui obiect dintr-o imagine, care prezintă avantajul unui timp mai redus de procesare comparativ cu algoritmul utilizat în paragraful precedent; organigrama corespunzătoare este prezentată în figura 3.23. Acest algoritm deşi oferă soluție după mai mulți paşi de procesări elementare, timpul total necesar nu depinde de dimensiunea obiectului la care i se caută punctul central. Pe de altă parte, algoritmul nu este afectat de erorile care se introduc în cazul în care obiectul nu este cu contur închis sau prezintă orificii. Algoritmul pentru determinarea punctului central al unui obiect dintr-o imagine se bazează pe identificarea într-o imagine cu niveluri de gri a poziției unui element de imagine dacă i se cunoaşte valoarea, fără a preciza numeric coordonatele acestuia în sistemul de referință ataşat planului de imagine.

Imaginea de intrare cu obiectul pentru care se determină punctul central trebuie să fie tot binară, ca în figura 3.24a. Cu convenția cunoscută, pixelii care aparțin obiectului, mulțimea **O**, au valoarea +1, celelalte elemente au valoarea –1:

$$(i,j) \in \mathbf{O} \mid \quad v(i,j) = +1, \tag{3.14}$$

$$(i,j) \notin \mathbf{O} \mid \quad v(i,j) = -1, \text{ pentru } 1 \le i \le M;\ 1 \le j \le N.$$

Acestei imagini binare i se asociază două imagini cu niveluri de gri ajutătoare, cu aceleaşi dimensiuni ca şi imaginea inițială de intrare. Pentru determinarea axei de

simetrie pe orizontală se va utiliza o imagine de "măsurare X" cu valori X (figura 3.24b), iar cealaltă imagine "măsurare Y" cu valori Y (figura 3.24c) serveşte pentru determinarea axei de simetrie pe verticală. În imaginea de "măsurare X", valorile pixelilor de pe o coloană sunt constante iar valorile coloanelor cresc liniar de la stânga la dreapta în intervalul de la –1 la +1. În imaginea de "măsurare Y", valorile pixelilor de pe o linie sunt constante şi valorile liniilor cresc de jos în sus, de la –1 la +1.

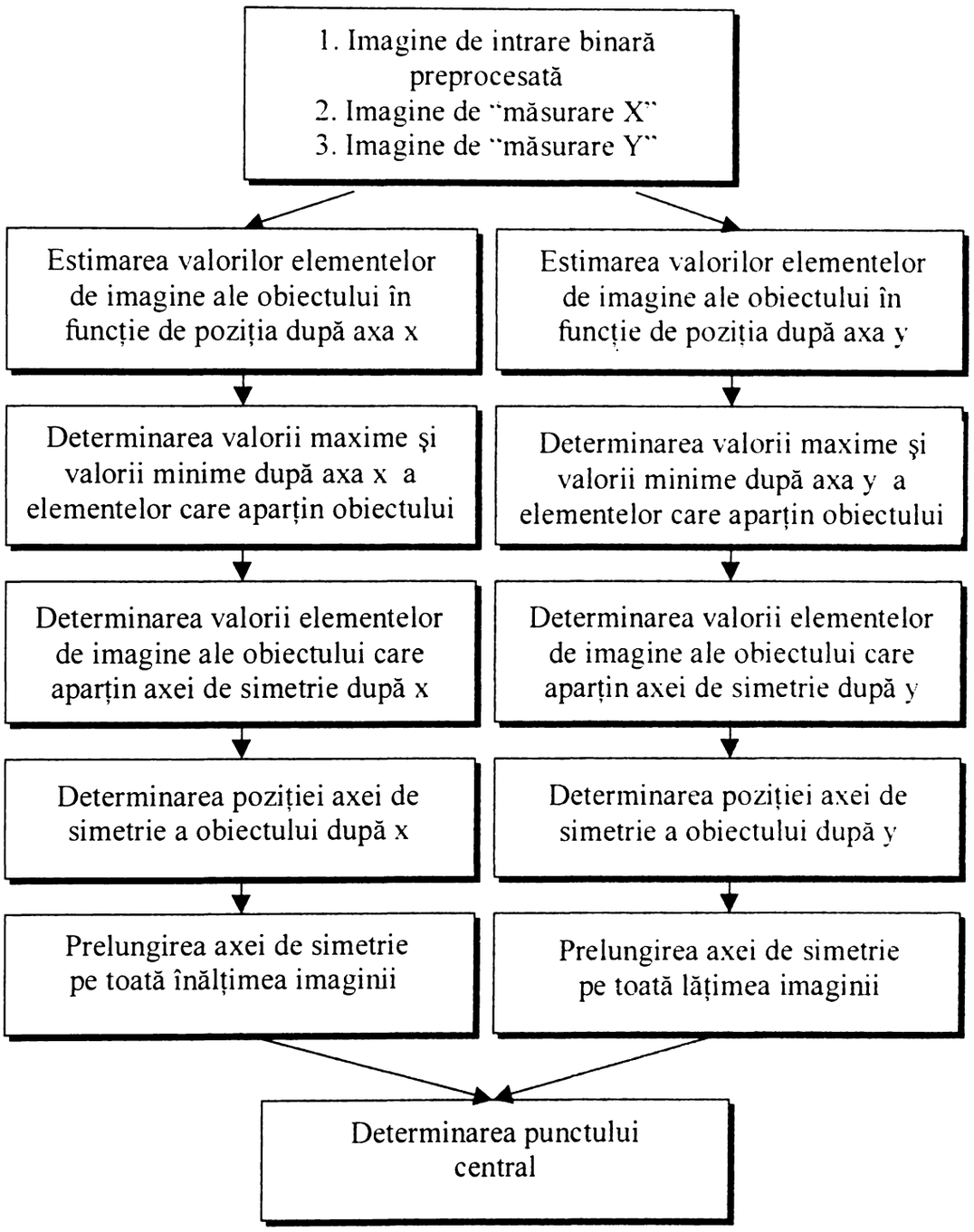


Figura 3.23. Organigrama noului algoritm CNN pentru determinarea punctului central al unui obiect.

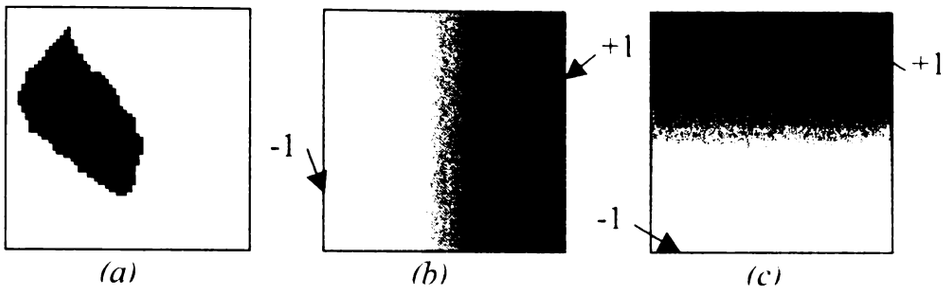


Figura 3.24. Imaginile de intrare care se utilizează la algoritmul CNN pentru determinarea punctului central al unui obiect: (a) imaginea binară cu un obiect; (b) imagine de "măsurare X"; (c) imagine de "măsurare Y".

Cu aceste două imagini de "măsurare" se realizează o evaluare după poziție a valorilor elementelor de imagine care aparțin numai obiectului. Imaginea binară de intrare se utilizează ca și mască (MASK), care nu permite modificarea valorilor anumitor elemente de imagine din imaginile de măsurare. De exemplu, se pot modifica numai valorile acelor elemente din imaginile de măsurare cărora le corespund în aceiași poziție în imaginea binară mască elemente active cu valoarea +1. Imaginea mască este inversa imaginii de intrare. Estimarea valorilor elementelor de imagine care aparțin obiectului, în funcție de poziție, se realizează pentru două situații:

- Dacă se utilizează operatorul filwhite.tem, relația (3.9) [125], valorile elementelor de imagine din imaginile de măsurare care nu aparțin obiectului în imaginea binară devin –1;

$$v(i,j)\big|_X = \begin{cases} X & \text{pentru } (i,j) \in \mathbf{O} \\ -1 & \text{pentru } (i,j) \notin \mathbf{O} \end{cases}, \tag{3.15}$$

$$v(i,j)\big|_Y = \begin{cases} Y & \text{pentru } (i,j) \in \mathbf{O} \\ -1 & \text{pentru } (i,j) \notin \mathbf{O} \end{cases}. \tag{3.16}$$

- Dacă se utilizează operatorul filblack.tem, relația (3.10) [125], valorile elementelor de imagine, din imaginile de măsurare care nu aparțin obiectului în imaginea binară devin +1;

$$v(i,j)\big|_X = \begin{cases} X & \text{pentru } (i,j) \in \mathbf{O} \\ +1 & \text{pentru } (i,j) \notin \mathbf{O} \end{cases}, \tag{3.17}$$

$$v(i,j)\big|_Y = \begin{cases} Y & \text{pentru } (i,j) \in \mathbf{O} \\ +1 & \text{pentru } (i,j) \notin \mathbf{O} \end{cases}. \tag{3.18}$$

Astfel, valorile elementelor din imaginile ajutătoare care aparțin obiectului depind de poziția pe care o are obiectul în imaginea binară.

În figura 3.25 se prezintă imaginile de ieşire după principalele etape de procesare ale algoritmului CNN pentru determinarea punctului central al unui obiect. După o succesiune de procesări elementare, accesibile cu instrucţiuni AMC, care folosesc imaginea binară de intrare şi imaginile de măsurare, se obţin valorile extreme ale elementelor din imaginea de măsurare care coincid în poziţie cu elementele obiectului în imaginea binară.

În acest fel, în imaginile din figura 3.25;a;c, devine posibilă determinarea valorilor maxime a elementelor care aparţin obiectului după cele două direcţii:

$$v_{max}(i,j)\big|_X \text{ şi } v_{max}(i,j)\big|_Y \text{ pentru } (i,j) \in \mathbf{O} \text{ şi } 1 \le i \le M;\ 1 \le j \le N. \qquad (3.19)$$

În imaginile din figura 3.25;b;d, se pot determina valorile minime a elementelor care aparţin obiectului, după cele două direcţii:

$$v_{min}(i,j)\big|_X \text{ şi } v_{min}(i,j)\big|_Y \text{ pentru } (i,j) \in \mathbf{O} \text{ şi } 1 \le i \le M;\ 1 \le j \le N. \qquad (3.20)$$

Valorile elementelor de imagine de pe axele de simetrie rezultă ca fiind media aritmetică dintre valoarea maximă şi cea minimă de pe direcţiile corespunzătoare (figura 3.25;g şi j).

Având determinate valorile elementelor de pe axele de simetrie poziţiile acestora se determină folosind tot imaginile ajutătoare din figura 3.25;a;b;c;d, printr-o succesiune de instrucţiuni AMC, elementare logice [130]. Punctul central se găseşte la intersecţia axelor de simetrie (figura 3.25k).

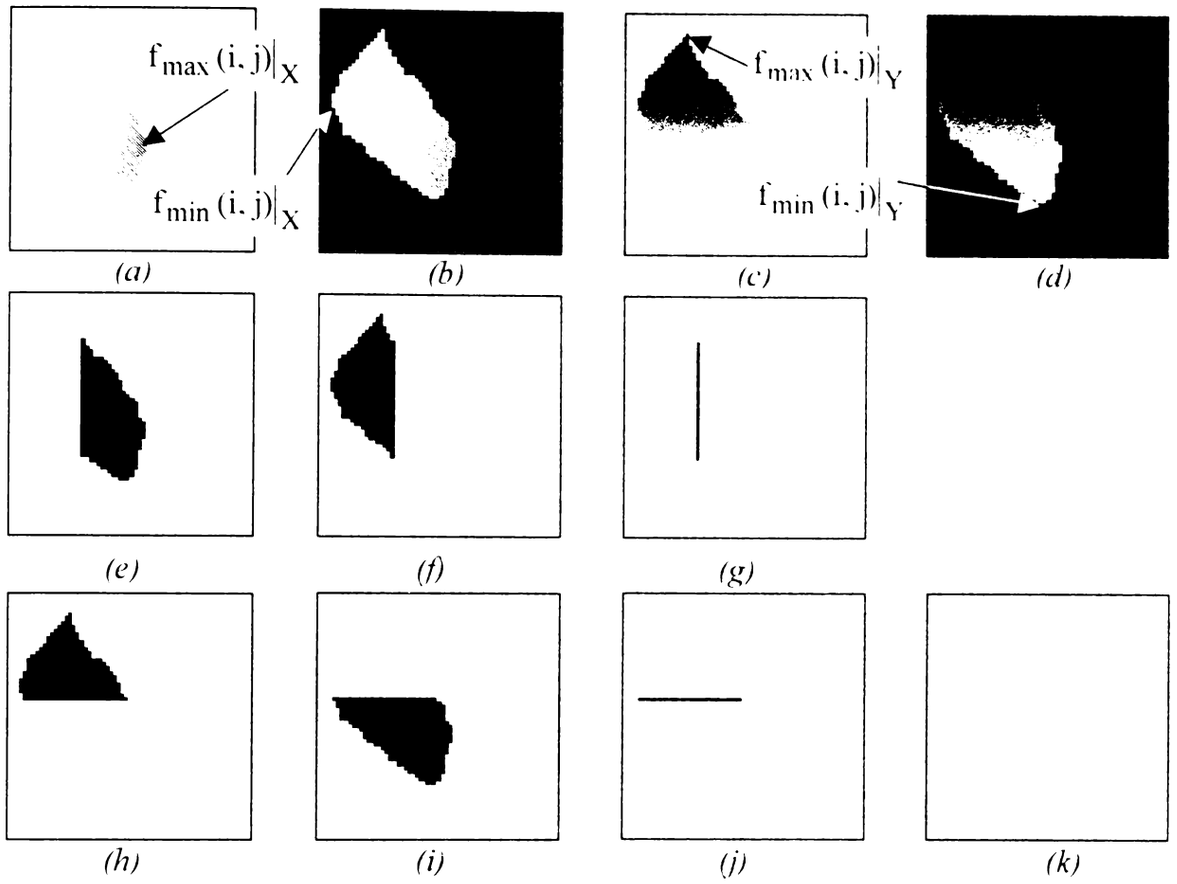


Figura 3.25. Imaginile de ieşire după principalele etape ale algoritmului CNN pentru determinarea punctului central al unui obiect: (a) şi (b) imagini utilizate la determinarea valorii maxime respectiv minime din obiect pe axa x; (c) şi (d) imagini utilizate la determinarea valorii maxime respectiv minime din obiect pe axa y; (e) şi (f) imagini utilizate la determinarea axei de simetrie după direcţia x; (h) şi (i) imagini utilizate la determinarea axei de simetrie după direcţia y; (g) şi (j) imagini care conţin axele de simetrie ale obiectului după direcţia x respectiv după direcţia y; (k) imagine care conţine punctul central al obiectului.

Prelungirea axelor de simetrie pe toată dimensiunea imaginii se realizează prin crearea unor umbre a acestor axe pe orizontală şi pe verticală, cu operatorii shsiud.tem, relaţia (3.11) şi shsirl.tem, relaţia (3.12) [125].

Aceste prelungiri ale axelor sunt necesare deoarece în cazul în care obiectul are orificii, punctul central al obiectului se poate situa şi în aceste orificii, aşa cum rezultă din figura 3.26.

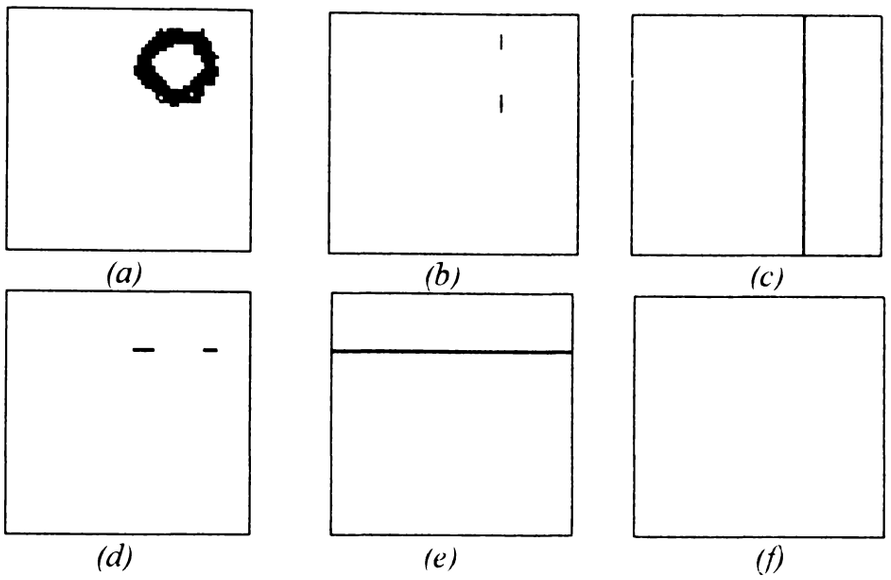


Figura 3.26. Determinarea punctului central al unui obiect cu orificii: (a) imaginea de intrare binară cu un obiect având orificii; (b) şi (d) imagini care conţin axele de simetrie ale obiectului după direcţia x respectiv după direcţia y; (c) şi (e) imagini care conţin axele de simetrie prelungite după direcţia x respectiv după direcţia y; (f) imagine care conţine punctul central al obiectului cu orificii.

Pentru testarea algoritmului prezentat mai sus s-a utilizat mediul de simulare "VisMouse-SimCNN" (Visual mouse platform and multi-layer CNN simulator [124]). În cazul unui obiect fără orificii, algoritmul CNN utilizează numai instrucţiuni elementare AMC (anexa 3), care nu necesită multe iteraţii. Pentru dimensiuni ale imaginii binare de intrare de 64*64 pixeli, operaţiile cu filwhite.tem şi filblack.tem, timpul necesar a fost de cel mult $10\tau_{CNN}$, celelalte procesări elementare fiind operaţii logice, fără tranzient. Dacă obiectul are însă orificii, timpul de procesare creşte datorită utilizării operatorilor shsiud.tem şi shsirl.tem. Chiar şi în acest caz, prin utilizarea algoritmului pentru determinarea punctului central al unui obiect dintr-o imagine, timpul total de procesare nu creşte proporţional cu dimensiunile obiectului şi nici cu dimensiunile imaginii; de aceea, această soluţie este mai avantajoasă faţă de utilizarea familiei de operatori center.tem.

3.6 Concluzii

În acest capitol au fost prezentați doi algoritmi originali, propuși de autor, pentru urmărirea unui obiect în mișcare, cu ajutorul unei camere video montate pe brațul unui robot și respectiv un algoritm de determinare a punctului central al unui obiect utilizând rețele neuronale celulare.

Algoritmul CNN de urmărire a unui obiect în mișcare cu ajutorul unei camere video mobile a fost verificat integral prin simulare și apoi testare pe un chip-CNN cu 20*22 de celule.

Pe baza rezultatelor obținute prin simulare și respectiv implementare pe un chip-CNN a algoritmului de urmărire a unui obiect în mișcare cu o cameră video, se poate considera că rețelele neuronale celulare constituie mijloace eficiente rapide și utile în rezolvarea comenzii vizuale bazate pe imagini a brațului robotic.

Între două imagini de intrare consecutiv achiziționate, prin prelucrarea complet paralelă cu rețele neuronale celulare a semnalelor bidimensionale, se asigură efectuarea tuturor procesărilor necesare în vederea funcționării în timp real a sistemului.

Soluțiile oferite pentru etapele algoritmului CNN de urmărire, permit implementarea imediată pe un chip-CNN de 64*64 de celule (cP4000 [68]), ori pe alte circuite CNN mai performante care vor apare în viitor. Se vor îmbunătăți astfel și performanțele algoritmului, deoarece timpul de prelucrare nu crește proporțional cu mărirea dimensiunii imaginilor prelucrate.

Algoritmul pentru determinarea punctului central al unui obiect utilizând rețele neuronale celulare se bazează numai pe procesări elementare logice și cu operatori liniari de dimensiune 3*3, fiind astfel posibilă implementarea cu ușurință a acestuia pe un chip-CNN de 64*64 de celule (cP4000 [68]). În acest fel, prelucrarea se poate efectua chiar în timp real, datorită procesării complet paralele și prin reducerea substanțială a numărului de încărcări pe chip a imaginilor inițiale.

Algoritmul menționat mai sus a fost testat integral prin simulare. Deși oferă soluție după mai mulți pași de procesări elementare, timpul total necesar nu depinde de dimensiunile obiectului ale cărui punct central urmează a fi determinat. Pe de altă parte, rezultatul conform algoritmului în discuție nu este afectat de erorile specifice obiectelor

al căror contur nu este închis sau care prezintă orificii, respectiv de erorile produse de necorelarea la start a direcției de decojire.

Utilizarea rețelelor neuronale celulare la identificarea poziției unui obiect dintr-o imagine, prin intermediul poziției punctului central, demonstrează încă odată că prin aceste metode de prelucrare se pot extrage proprietăți globale din imagini, chiar dacă procesările CNN sunt aparent numai locale.

4. Interpolarea semnalelor bidimensionale utilizând rețele neuronale celulare

În acest capitol sunt prezentate noi metode de interpolare, propuse de autor, care utilizează rețele neuronale celulare, cu operatori liniari de reacție de dimensiune 3*3, obținuți pe cale analitică. Prin implementarea acestor metode de interpolare pe un chip-CNN se realizează o procesare complet paralelă şi deci în timp real [68].

Interpolarea CNN a semnalelor bidimensionale se poate utiliza la obținerea semnalelor bidimensionale de comandă pentru roboți. Totodată, mărirea imaginii prin interpolare cu păstrarea dimensiunilor elementelor de imagine poate fi utilă la interpretarea imaginilor biomedicale respectiv la obținerea unor imagini de calitate HDTV recepționând semnalul TV analogic uzual [121]. Sunt prezentate, pe parcursul acestui capitol, rezultatele verificării prin simulare şi testare pe un chip-CNN a acestor metode de interpolare în cazul creşterii rezoluției imaginilor binare şi cu niveluri de gri, la codarea şi decodarea imaginilor în vederea transmiterii lor şi la reconstrucția de imagini.

4.1 Interpolarea semnalelor bidimensionale

Semnalele bidimensionale de interpolat se consideră ca fiind funcții discrete bidimensionale, $v(x_i,y_j)$, de dimensiune M*N. Se va folosi termenul de imagine pentru toate semnalele bidimensionale chiar dacă natura acestora nu este de imagine propriu-zisă (spre exemplu, "imaginile" de comandă ale roboților sau "imaginile" care provin de la senzorii tactili ai acestora). Variabilele (x_i,y_j), $1 \le i < M$ şi $1 \le j < N$, reprezintă coordonatele spațiale pe orizontală respectiv verticală ale unui element de imagine oarecare. Valorile funcției $v(x_i,y_j)$ pentru imagini cu niveluri de gri sunt în domeniul standard CNN adică [-1, +1], de la alb către negru, iar pentru imagini binare aceste valori sunt +1 pentru negru şi −1 pentru alb. Aceste valori sunt interpretate ca înălțimi care descriu o suprafață în raport cu o suprafață de referință.

Dacă la un moment dat se cunosc valorile elementelor de imagine $v(x_i,y_j)$ pentru un număr P finit de elemente numite noduri, interpolarea imaginilor constă în estimarea valorile elementelor de imagine necunoscute, din valorile elementelor de imagine care se cunosc. Pentru estimarea funcției $v(x_i,y_j)$ în punctul în care valorile acesteia nu se

cunosc se poate folosi o funcție de interpolare $V(x_i,y_j)$ care satisface condițiile [45,83,96]:

$$v(x_i,y_j)=V_p, \text{ unde } p=1,2,\ldots,P. \tag{4.1}$$

Funcția $V(x_i,y_j)$ se obține dintr-un set de funcții liniar independente $\vartheta(x_i,y_j)$:

$$V(x_i,y_j)=\sum_{p=1}^{P}\alpha_p\vartheta(x_i,y_j), \tag{4.2}$$

unde α_p sunt soluțiile sistemului:

$$\sum_{p=1}^{P}\alpha_p\vartheta(x_i,y_j)=V_p \tag{4.3}$$

Dacă se utilizează, de exemplu, ca funcții liniar independente şirul funcțiilor trigonometrice, se obține chiar seria Fourier [96]. De cele mai multe ori însă se alege un polinom algebric ca funcție de interpolare, $V(x_i,y_j)$. Indiferent de metoda de interpolare care se alege, se doreşte ca imaginea care rezultă să aibă erori cât mici față de o imagine considerată de referință.

În scopul simplificării notării se va folosi, dacă este cazul, convenția:

$v(x_i,y_j)=v(i,j)$, pentru $1 \le i < M$ şi $0 \le j < N$.

Interpolarea nu este identică cu aproximarea unei suprafețe. Suprafața care rezultă prin interpolare trece neapărat prin toate punctele care au fost considerate cunoscute la începutul procesării în timp ce în cazul aproximării această condiție nu se realizează. Astfel, prin aproximare se pot elimina erorile grosolane care ar putea exista printre valorile unor noduri obținute ca rezultate experimentale, care evident se cunosc cu o anumită incertitudine.

Un alt aspect deosebit de important privind interpolarea îl constituie timpul de calcul, care trebuie să fie cât mai scurt, astfel încât să existe posibilitatea execuției aplicației în timp real. Prin implementarea pe un chip-CNN a metodelor de interpolare CNN propuse, procesările sunt complet paralele şi se pot realiza în timp real.

Prin utilizarea rețelelor neuronale celulare creşte viteza de procesare, însă folosirea directă a unor funcții de interpolare nu este posibilă. De aceea, la proiectarea de operatori vom avea în vedere minimizarea unor norme care vor asigura ca suprafețele descrise de valorile, amplitudinile elementelor de imagine, să fie cât mai netede. Operatorii care rezultă vor avea un efect similar interpolării cu funcții liniare, pătratice sau spline cubice.

Imaginile de intrare utilizate la testarea metodelor de interpolare prin simulare sau procesare pe chip, diferă între ele după cum sunt dispuse elementele de imagine cu valori necunoscute față de elementele de imagine ale căror valori se cunosc. La toate imaginile de intrare care se vor interpola, se consideră cunoscute şi poziţiile elementelor de imagine care au valori cunoscute. Valorile acestor elemente nu se vor modifica în decursul procesului tranzitoriu datorită utilizării unor imagini mască. Se vor avea în vedere următoarele situaţii:

i.) Mărirea dimensiunilor imaginii, adică se intercalează pe linii şi pe coloane noi elemente de imagine față de elementele de imagine ale căror valori se cunosc. Creşterea dimensiunii pe orizontală şi pe verticală a imaginii se face cu păstrarea dimensiunilor elementelor de imagine.

ii.) Reconstrucţia unei imagini oarecare. În acest caz se cunosc valorile doar la o parte din elementele unei imagini, dispuse neregulat şi se necesită obţinerea prin calcul a elementelor lipsă. Imaginea de ieşire are aceleaşi dimensiuni şi rezoluţie ca şi imaginea de intrare.

iii.) Reconstrucţia unei imagini de intrare la care elementele cu valori necunoscute sunt dispuse regulat pe linii şi coloane. Şi în acest caz rezoluţia imaginii de ieşire rămâne identică cu rezoluţia imaginii de intrare, la fel şi dimensiunea.

Ultimele două situaţii permit o evaluare uşoară a erorii care rezultă în urma interpolării cu metodele propuse, deoarece imaginile se obţin din imagini reale la care s-au anulat valorile corespunzătoare unei părţi din elementele imaginii.

iv.) Interpolare pentru transmiterea la distanţă a unor imagini. La emisie are loc reducerea dimensiunii imaginilor prin codare iar la recepţie prin decodare se reface imaginea cu dimensiunea şi rezoluţia iniţială.

4.2 Mărirea imaginilor cu păstrarea dimensiunilor elementului de imagine

4.2.1 Principiul interpolării liniare

Mărirea imaginilor cu păstrarea dimensiunilor elementelor de imagine se poate realiza prin interpolare. În principiu, mărirea imaginilor constă în creşterea numărului de linii şi de coloane. Se introduc noi elemente de imagine ale căror valori sunt calculate prin interpolare [42,55]. Înaintea interpolării, propriu-zise, se va mări deci dimensiunea imaginii iniţiale descrisă de variabila generală v(i,j), prin inserţie de elemente de imagine pe linii şi coloane cu valori nule.

În figura 4.1 se prezintă principiul de interpolare pentru cazul procesării secvenţiale, la creşterea dimensiunii unei imagini de 2 ori pe linii şi de 2 ori pe coloane. Pixelii de culoarea albă reprezintă elementele iniţiale, la care se cunosc valorile înainte de interpolare, iar cele de culoarea gri sunt elemente cu valori necunoscute, care se calculează prin interpolare.

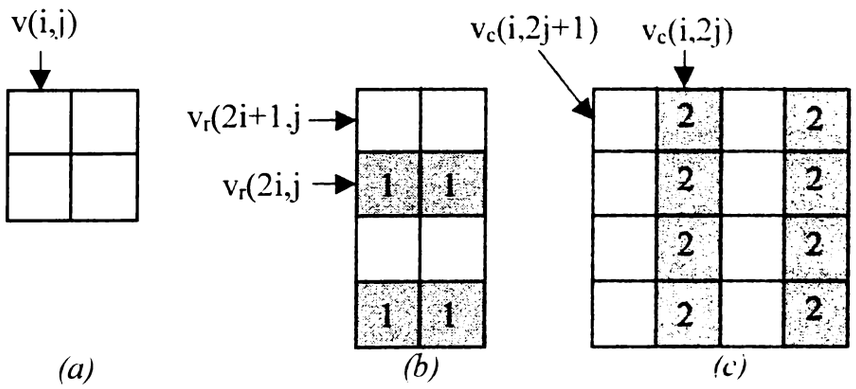


Figura 4.1. Creşterea dimensiunilor imaginii prin interpolare, cu păstrarea dimensiunilor elementului de imagine: (a) imaginea iniţială; (b) imaginea care se obţine în urma inserţiei de linii; (c) imaginea care se obţine în urma inserţiei de coloane.

Dacă se utilizează metoda de interpolare liniară, valorile elementelor de imagine obţinute prin inserare (zerourile) se vor înlocui cu media aritmetică a valorilor vecinilor acestora, calculul efectuându-se în două etape, după cum urmează:

1.) în prima etapă se calculează pe linii elementele notate cu 1, element după element, ca în figura 4.1b:

$$v_r(2i,j)=\frac{[v(i,j)+v(i+1,j)]}{2}, \tag{4.4}$$

$$v_r(2i+1,j)=v(i,j);\quad 0\le i\le M-1, 0\le j\le N-1; \tag{4.5}$$

2.) în a doua etapă, se calculează elementele pe coloane notate cu 2, element după element (figura 4.1c):

$$v_c(i,2j)=\frac{[v(i,j)+v(i,j+1)]}{2}, \tag{4.6}$$

$$v_c(i,2j+1)=v(i,j);\quad 0\le i\le M-1, 0\le j\le N-1. \tag{4.7}$$

În expresiile de mai sus, elementele de imagine $v_r(2i,j)$, $v_r(2i+1,j)$, $v_c(i,2j)$, $v_c(i,2j+1)$ corespund imaginii mărite pe linii şi pe coloane, iar $v(i,j)$, $v(i,j+1)$, $v(i+1,j)$ sunt elemente de imagine ale imaginii inițiale.

În cazul utilizării metodei de interpolare descrise mai sus, datorită principiului secvențial şi volumului mare de calcul, timpul de procesare este ridicat şi creşte proporțional odată cu dimensiunea imaginii.

4.2.2 Interpolarea liniară a imaginilor utilizând rețele neuronale celulare

Pentru mărirea imaginii, cu păstrarea dimensiunilor elementului de imagine, se prezintă un algoritm de interpolare liniară cu implementare CNN în lucrarea [108]. Prin această metodă rezultatul final se obține în două etape, în fiecare dintre aceste etape utilizându-se numai operatori de comandă, de tip B (figura 4.2). Pixelii de culoarea albă reprezintă elementele la care se cunosc valorile la începutul interpolării. În prima etapă se calculează valorile elementelor notate cu 1 cu ajutorul operatorului B1, iar în a doua etapă se obțin valorile elementelor notate cu 2 cu ajutorul operatorului B2. Prin utilizarea rețelelor neuronale celulare pentru interpolare, în fiecare etapă procesarea se poate realiza paralel. În cazul interpolării liniare CNN, operatorii de comandă sunt definiți de relațiile [108]: (4.8)

$$B1=\begin{array}{|c|c|c|}\hline 0.25 & 0 & 0.25\\ \hline 0 & 0 & 0\\ \hline 0.25 & 0 & 0.25\\ \hline\end{array} \qquad B2=\begin{array}{|c|c|c|}\hline 0 & 0.25 & 0\\ \hline 0.25 & 0 & 0.25\\ \hline 0 & 0.25 & 0\\ \hline\end{array}$$

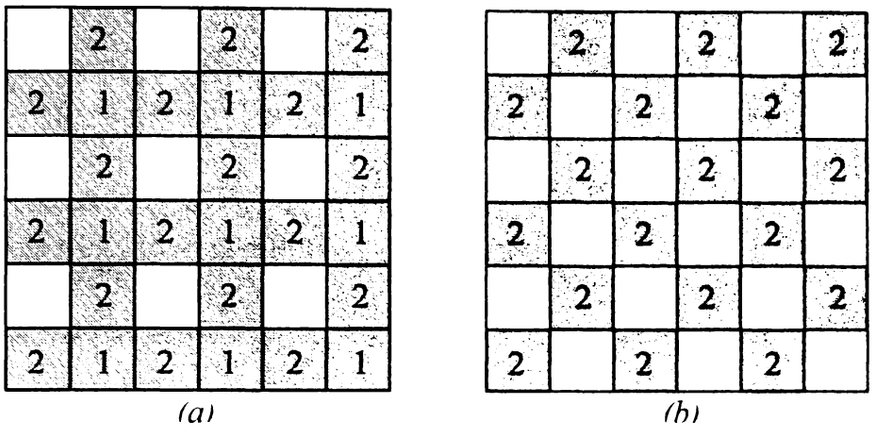


Figura 4.2. Interpolare CNN cu operator de tip B: (a) imaginea inițială inserată cu zerouri pe linii şi coloane, la etapa determinării prin interpolare cu operatorul B1 a elementelor de imagine notate cu 1; (c) determinarea prin interpolare cu operatorul B2 a elementelor de imagine notate cu 2.

Pentru a se obține în urma interpolării un comportament mai bun la muchii, se poate utiliza un alt algoritm adaptiv CNN de interpolare descris în lucrarea [121], care foloseşte numai operatori de comandă. Interpolarea este efectuată tot în două etape, cu operatori de tip B asemănători cu operatorii B1 şi B2 din exemplul precedent. În acest algoritm, pentru a calcula valoarea necunoscută a unui element de imagine prin interpolare liniară dintre cei 4 vecinii cunoscuți, se aleg numai acei doi de pe o diagonală, pentru care diferența dintre valorile elementelor este mai mică. În acest fel, valoarea elementului calculat nu va deteriora muchia.

4.3 Interpolarea imaginilor utilizând rețele neuronale celulare cu operator de reacție

În cazul utilizării la interpolare a rețelelor neuronale celulare se are în vedere faptul că deşi elementele sau celulele nu sunt conectate decât local, prin conexiuni limitate de vecinătăți de rază r, totuşi datorită efectului de propagare determinat de comportamentul dinamic al celulelor, starea respectiv ieşirea unei celule va fi influențată şi de celulele din afara propriei vecinătăți directe, N_r. Acest efect are loc numai dacă există o reacție de la ieşirea celulelor vecine la starea celulei, prin intermediului operatorului de reacție A. Dacă se implementează pe un chip-CNN metoda de interpolare realizată numai cu operator de reacție, procesarea devine complet paralelă.

În lucrarea [19] se propune un operator de tip A, de dimensiune 5*5. Însă acest operator nu poate fi implementat pe circuitele CNN existente, atât datorită dimensiunii prea mari cât şi a valorilor parametrilor din operator.

În cele ce urmează sunt prezentate metodele noi, propuse de autor, de interpolare a imaginilor utilizând rețele neuronale celulare. Totodată, sunt descrise procedeele de dimensionare a operatorilor liniari de reacție de dimensiune 3*3, cu care se realizează interpolarea CNN a imaginilor.

4.3.1 Condiții inițiale de proiectare pentru determinarea operatorilor A de reacție utilizați la interpolare

În general, la proiectarea în domeniul CNN, un grad sporit de dificultate îl prezintă determinarea operatorilor de reacție de tip A, utilizați la prelucrări de imagini cu niveluri de gri. În continuare sunt descrise condițiile inițiale utilizate la proiectarea acestui tip de operator:

1.) În cazul interpolării CNN utilizând operator cu reacție este neapărat necesară utilizarea unei imagini mască care să nu permită modificarea valorilor celulelor care se cunosc la începutul interpolării. Elementele care nu trebuie să-şi modifice starea în decursul procesului tranzitoriu corespund pozițiilor pentru care pixelii din mască au o anumită valoarea, de exemplu, -1.

2.) La proiectarea de operatori cu reacție, indiferent de metoda utilizată, se va avea în vedere să se elimine necesitatea cunoașterii condițiilor de frontieră. Acest deziderat se poate realiza prin prelungirea imaginii la frontiere, cu linii și coloane virtuale, care repetă valorile celulelor vecine [13,62]. În cazul procesării CNN această condiție de frontieră este de tipul zero-flux.

3.) Imaginea care urmează a se interpola cu zerouri inserate, se aplică pe starea STATE a rețelei, deci la momentul t=0, x_{ij}= v(i,j). Evoluția fiecărei celule interne C_{ij}, este caracterizată de ecuația de stare:

$$\dot{x} = \frac{dx_{ij}}{dt} = -x_{ij} + \sum_{C_{kl} \in N_r} A_{ij,kl} y_{kl} + \sum_{C_{kl} \in N_r} B_{ij,kl} u_{kl} + z_{ij}, \qquad (4.9)$$

pentru $1 \le i \le M; \quad 1 \le j \le N$.

În relația de mai sus u_{kl} reprezintă intrarea, x_{ij} reprezintă starea iar y_{kl} reprezintă ieșirea. Celulele C_{kl} sunt din vecinătatea de rază r a celulei C_{ij}, adică $C_{kl} \in N_r$. Operatorul de reacție s-a notat cu $A_{ij,kl}$, $B_{ij,kl}$ semnifică operatorul de comandă iar z_{ij} reprezintă polarizarea.

În cele ce urmează se vor folosi numai operatori de reacție de tip A iar pe intrarea rețelei se va aplica orice imagine. Operatorul de comandă de tip B respectiv polarizarea z se consideră nule, adică:

$$\sum_{C_{kl} \in N_r} B_{ij,kl} u_{kl} = 0 \quad \text{și} \quad z_{ij} = 0. \qquad (4.10)$$

4.) Operatorul A se presupune de forma:

$$A = \begin{bmatrix} a_{-1,-1} & a_{-1,0} & a_{-1,+1} \\ a_{0,-1} & a_{0,0} & a_{0,+1} \\ a_{+1,-1} & a_{+1,0} & a_{+1,+1} \end{bmatrix}. \qquad (4.11)$$

Pentru ca această configurație de operator să asigure stabilitatea rețelei și menținerea în orice situație a valorilor de ieșire y_{ij} în domeniul liniar [-1,1], condiția necesară și suficientă care trebuie să se îndeplinească este ca:

$$\left| \sum_{C_{kl} \in N_r} A_{ij,kl} y_{kl} \right| \le 1, \text{ pentru } 1 \le i \le M; \quad 1 \le j \le N. \qquad (4.12)$$

În acest caz ieșirea rețelei este identică cu starea rețelei, adică:

$$y_{ij} \equiv x_{ij}, \text{ pentru } 1 \le i \le M; \quad 1 \le j \le N. \qquad (4.13)$$

5.) La dimensionarea operatorului de reacție se are în vedere condiția de a obține în final o stare stabilă x_{ij}^*, pentru fiecare celulă. Pentru ca rețeaua să ajungă într-o stare stabilă trebuie să se îndeplinească condiția [14]:

$$\lim_{t\to\infty} x_{ij}(t) = x_{ij}^*, \text{ pentru } 1 \le i \le M; \quad 1 \le j \le N, \tag{4.14}$$

respectiv:

$$\left.\frac{dx_{ij}}{dt}\right|_{x_{ij}=x_{ij}^*} = 0. \tag{4.15}$$

Totodată din relațiile (4.9), (4.10), (4.11) (4.13), şi (4.15) rezultă:

$$\sum_{C_{kl}\in N_r} A_{ij,kl}y_{kl} = x_{i,j} = y_{i,j}, \text{ respectiv:} \tag{4.16}$$

$$\sum_{C_{kl}\in N_r} A_{ij,kl}y_{kl} = a_{-1,-1}y_{i-1,j-1} + a_{-1,+1}y_{i-1,j+1} + a_{+1,+1}y_{i+1,j+1} + a_{+1,-1}y_{i+1,j-1} +$$

$$+ a_{-1,0}y_{i-1,j} + a_{+1,0}y_{i+1,j} + a_{0,-1}y_{i,j-1} + a_{0,-1}y_{i,j+1} + a_{0,0}y_{i,j} = y_{i,j}$$

6.) La proiectarea de operatori cu reacție se pot utiliza şi metode de minimizare a unor norme care pot constitui funcții de cost sau de energie. Prin găsirea minimelor globale pentru aceste funcții de cost se pot dimensiona şi alți operatori de reacție, care asigură o interpolare pătratică sau spline cubică. Asemenea funcții de cost, pentru care se calculează punctele de minim sunt prezentate în lucrările [45,46,102].

Astfel, pentru exemplificare, în cazul interpolării unei funcții unidimensionale v(x), se are în vedere minimizarea normei:

$$v_m^2 = \int_a^b \left|\frac{d^m v(x)}{dx^m}\right|^2 dx, \quad \text{cu} \quad P \ge m \ge 1, \tag{4.17}$$

unde P este numărul de puncte ale variabilei x pentru care se cunosc valorile v(x), adică $v(x)=V_p$ unde p=1,2,...,P.

În cazul interpolării unei funcții unidimensionale se recomandă două variante practice de norme [46,102]:

modelul de fir infinit de lung pentru m=1,

$$|v|^2 = \int_a^b \left|\frac{dv}{dx}\right|^2 dx, \tag{4.18}$$

modelul de bară elastică infinită pentru m=2,

$$|v|^2 = \int_a^b \left|\frac{d^2v}{dx^2}\right|^2 dx . \qquad (4.19)$$

Pentru cazul semnalelor bidimensionale, vor rezulta ca modele de interpolare modelul de membrană (m=1) şi modelul de placă subţire (m=2). Relaţiile corespunzătoare sunt prezentate în continuare [46,102]:

$$E = \iint_{x,y\in D}\left[(\frac{\partial v}{\partial x})^2 + (\frac{\partial v}{\partial y})^2\right]dxdy . \qquad (4.20)$$

$$(x,y)\in D \mid \quad 1\le x\le M;\quad 1\le y\le N ,$$

$$E(v(x,y)) = \iint_{x,y\in D}\left[(\frac{\partial^2 v}{\partial x^2})^2 + 2(\frac{\partial^2 v}{\partial x\partial y})^2 + (\frac{\partial^2 v}{\partial y^2})^2\right]dxdy . \qquad (4.21)$$

Dintre toate funcţiile care aproximează funcţia $v(x_i,y_j)$ între nodurile:

$v(x_i,y_j)=V_p$; p=1,2,....P, unde P este numărul de puncte ale variabilei v pentru care se cunosc valorile, funcţiile spline cubice au proprietatea remarcabilă de a minimiza integrală din expresia (4.21)[96].

Minimizarea acestor funcţii de cost vor asigura ca suprafeţele descrise de valorile elementelor de imagine să fie cât mai netede.

În urma discretizării spaţiale cu un pas de eşantionare h, identic pe linii şi pe coloane, adică:

$$x_{i+1} - x_i = x_i - x_{i-1} = h \qquad (4.22)$$

$$y_{j+1} - y_j = y_j - y_{j-1} = h ,$$

integralele (4.20) şi (4.21) se vor transforma în sume.

Se va omite în continuare scrierea completă a variabilelor spaţiale x_i şi y_j pentru a nu se confunda cu starea şi respectiv ieşirea reţelei. Se va folosi convenţia $v(x_i,y_j) \equiv v_{i,j}$.

Conform celor de mai sus, derivatele parţiale într-un punct (i,j) pentru funcţia $v_{i,j}$ sunt date de relaţiile: (4.23)

$$\frac{\partial v_{i,j}}{\partial x} = \frac{v_{i+1,j} - v_{i,j}}{h} ,$$

$$\frac{\partial v_{i,j}}{\partial x} = \frac{v_{i,j} - v_{i-1,j}}{h} ,$$

$$\frac{\partial v_{i,j}}{\partial y} = \frac{v_{i,j+1} - v_{i,j}}{h},$$

$$\frac{\partial v_{i,j}}{\partial y} = \frac{v_{i,j} - v_{i,j-1}}{h},$$

$$\frac{\partial^2 v_{i,j}}{\partial x^2} = \frac{v_{i+1,j} - v_{i,j} - v_{i,j} + v_{i-1,j}}{h^2},$$

$$\frac{\partial^2 v_{i,j}}{\partial y^2} = \frac{v_{i,j+1} - v_{i,j} - v_{i,j} + v_{i,j-1}}{h^2},$$

$$\frac{\partial^2 v_{i,j}}{\partial x \partial y} = \frac{1}{h^2}(v_{i+1,j+1} - v_{i+1,j} - v_{i,j+1} + v_{i,j}),$$

$$\frac{\partial^2 v_{i,j}}{\partial x \partial y} = \frac{1}{h^2}(v_{i,j} - v_{i,j-1} - v_{i-1,j} + v_{i-1,j-1}),$$

$$\frac{\partial^2 v_{i,j}}{\partial x \partial y} = \frac{1}{h^2}(v_{i,j+1} - v_{i-1,j+1} - v_{i,j} + v_{i-1,j}),$$

$$\frac{\partial^2 v_{i,j}}{\partial x \partial y} = \frac{1}{h^2}(v_{i+1,j} - v_{i,j} - v_{i+1,j-1} + v_{i,j-1}).$$

4.3.2 Interpolare CNN liniară utilizând operator de reacție

Se consideră că operatorul care va rezulta este simetric în raport cu celula centrală, adică în matricea A elementele satisfac condițiile:

$$a_{-1,-1} = a_{-1,0} = a_{-1,+1} = a_{0,-1} = a_{0,+1} = a_{+1,-1} = a_{+1,0} = a_{+1,+1} = a, \quad (4.24)$$

$a_{0,0} = 0$, de asemenea, (4.25)

se mai impune condiția de aproximare finală:

$$y_{i-1,j-1} \cong y_{i-1,j+1} \cong y_{i+1,j+1} \cong y_{i+1,j-1} \cong y_{i-1,j} \cong y_{i+1,j} \cong y_{i,j-1} \cong y_{i,j+1} = y. \quad (4.26)$$

Din relațiile de mai sus rezultă: a=0.125. (4.27)

Operatorul de reacție aintpol1.tem care asigură interpolare liniară este de forma:

(4.28)

A =

0.125	0.125	0.125
0.125	0	0.125
0.125	0.125	0.125

B =

0	0	0
0	0	0
0	0	0

z = | 0 |

Acest operator asigură, în starea stabilă a rețelei, ca ieșirea fiecărei celule necunoscute să fie egală cu media valorilor vecinilor săi.

4.3.3 Interpolare CNN pătratică utilizând operator de reacție

Funcția de cost care rezultă prin discretizarea relației (4.20), pentru m=1, căreia trebuie să i se găsească un minim global pentru interpolarea imaginii $v(x_i,y_j)$ este:

$$E(v(x_i,y_j)) = \sum_{i=1}^{M}\sum_{j=1}^{N}\left[\left(\frac{\partial v}{\partial x_i}\right)^2 + \left(\frac{\partial v}{\partial y_j}\right)^2\right]. \tag{4.29}$$

Se va folosi o altă formă convenabilă a expresiei de mai sus, obținută pe baza relațiilor (4.23):

$$E(v_{i,j}) = \sum_{i=1}^{M}\sum_{j=1}^{N}\frac{1}{h^2}\left[(v_{i+1,j}-v_{i,j})(v_{i,j}-v_{i-1,j}) + (v_{i,j+1}-v_{i,j})(v_{i,j}-v_{i,j-1})\right]. \tag{4.30}$$

Valorile variabilei v, pentru care E este minim, se obțin din condiția:

$$\frac{\partial E(v)}{\partial v_{i,j}} = 0;\ \text{pentru } 1 \le i \le M;\quad 1 \le j \le N. \tag{4.31}$$

Se exprimă în continuare funcția de cost în raport cu ieșirea rețelei:

$$E(v) = E(y),\ \text{respectiv} \tag{4.32}$$

$$E(y) = \sum_{i=1}^{M}\sum_{j=1}^{N}\frac{1}{h^2}\left[(y_{i+1,j}-y_{i,j})(y_{i,j}-y_{i-1,j}) + (y_{i,j+1}-y_{i,j})(y_{i,j}-y_{i,j-1})\right]. \tag{4.33}$$

$$\begin{aligned} E(y) = \sum_{i=1}^{M}\sum_{j=1}^{N}\frac{1}{h^2}\big(&y_{i+1,j}y_{i,j} - y_{i,j}^2 - y_{i+1,j}y_{i-1,j} + y_{i,j}y_{i-1,j} + \\ &+ y_{i,j+1}y_{i,j} - y_{i,j}^2 - y_{i,j+1}y_{i,j-1} + y_{i,j}y_{i,j-1}\big) \end{aligned} \tag{4.34}$$

Pentru a găsi minimul acestei funcții de cost trebuie ca:

$$\frac{\partial E}{\partial y_{i,j}} = 0;\ \text{pentru } 1 \le i \le M;\quad 1 \le j \le N,\ \text{adică:} \tag{4.35}$$

$$\frac{\partial E}{\partial y_{i,j}} = \sum_{i=1}^{M}\sum_{j=1}^{N}\frac{1}{h^2}(y_{i+1,j} - 2y_{i,j} + y_{i-1,j} + y_{i,j+1} - 2y_{i,j} + y_{i,j-1}) = 0, \tag{4.36}$$

$$4y_{i,j} = y_{i-1,j} + y_{i,j+1} + y_{i,j-1} + y_{i+1,j},$$

Din egalitatea de mai sus rezultă:

$$y_{i,j} = \frac{y_{i-1,j} + y_{i,j+1} + y_{i,j-1} + y_{i+1,j}}{4}. \tag{4.37}$$

Conform relației (4.16) se poate scrie:

$$\sum_{C_{kl} \in N_r} A_{ij,kl} y_{kl} = a_{-1,-1} y_{i-1,j-1} + a_{-1,+1} y_{i-1,j+1} + a_{+1,+1} y_{i+1,j+1} + a_{+1,-1} y_{i+1,j-1} + \tag{4.38}$$
$$+ a_{-1,0} y_{i-1,j} + a_{+1,0} y_{i+1,j} + a_{0,-1} y_{i,j-1} + a_{0,-1} y_{i,j+1} + a_{0,0} y_{i,j} = y_{i,j}$$

Prin identificarea coeficienților între expresiile (4.37) și (4.38) rezultă elementele operatorului A:

$$a_{-1,-1} = a_{-1,+1} = a_{+1,-1} = a_{+1,+1} = 0, \tag{4.39}$$

$$a_{-1,0} = a_{0,-1} = a_{0,+1} = a_{+1,0} = 0.25,$$

$$a_{0,0} = 0.$$

În consecință, operatorul de reacție aintpol2.tem, care asigură interpolare pătratică, este de forma: (4.40)

$$A = \begin{array}{|c|c|c|}\hline 0 & 0.25 & 0 \\ \hline 0.25 & 0 & 0.25 \\ \hline 0 & 0.25 & 0 \\ \hline \end{array} \qquad B = \begin{array}{|c|c|c|}\hline 0 & 0 & 0 \\ \hline 0 & 0 & 0 \\ \hline 0 & 0 & 0 \\ \hline \end{array} \qquad z = \boxed{0}$$

4.3.4 Interpolare CNN spline cubică utilizând operator de reacție

Funcția de cost care rezultă prin discretizarea relației (4.21), pentru m=2, căreia trebuie să se găsească un minim global pentru interpolarea imaginii $v(x_i,y_j)$ este:

$$E(v(x_i, y_j)) = \sum_{i=1}^{M} \sum_{j=1}^{N} \frac{1}{h^2} \left[\left(\frac{\partial^2 v}{\partial x^2}\right)^2 + 2\left(\frac{\partial^2 v}{\partial x \partial y}\right)^2 + \left(\frac{\partial^2 v}{\partial y^2}\right)^2 \right] \tag{4.41}$$

Se va folosi o altă formă convenabilă a expresiei de mai sus, obținută pe baza relațiilor (4.23):

$$E(v_{i,j}) = \sum_{i=1}^{M} \sum_{j=1}^{N} \frac{1}{h^2} \Big[(v_{i-1,j} - 2v_{i,j} + v_{i+1,j})^2 (v_{i,j-1} - 2v_{i,j} + v_{i,j+1})^2 -$$
$$- (v_{i+1,j+1} - v_{i+1,j} - v_{i,j+1} + v_{i,j})(v_{i,j} - v_{i,j-1} - v_{i-1,j} + v_{i-1,j-1}) + \tag{4.42}$$
$$(-1)(v_{i-1,j+1} + v_{i,j} - v_{i,j+1} - v_{i-1,j})(-1)(v_{i,j} + v_{i+1,j-1} - v_{i,j-1} - v_{i+1,j}) \Big]$$

Valorile variabilei v, pentru care E este minim, se obțin din condiția:

$$\frac{\partial E}{\partial v_{i,j}} = 0 ; \text{ pentru } 1 \le i \le M; \quad 1 \le j \le N . \tag{4.43}$$

Se exprimă în continuare funcția de cost în raport cu ieșirea rețelei:

$E(v) = E(y)$ respectiv: (4.44)

$$E(y) = \sum_{i=1}^{M}\sum_{j=1}^{N}\frac{1}{h^2}\Big[(y_{i-1,j} - 2y_{i,j} + y_{i+1,j})^2 (y_{i,j-1} - 2y_{i,j} + y_{i,j+1})^2 -$$

$$-(y_{i+1,j+1} - y_{i+1,j} - y_{i,j+1} + y_{i,j})(y_{i,j} - y_{i,j-1} - y_{i-1,j} + y_{i-1,j-1}) + \tag{4.45}$$

$$(-1)(y_{i-1,j+1} + y_{i,j} - y_{i,j+1} - y_{i-1,j})(-1)(y_{i,j} + y_{i+1,j-1} - y_{i,j-1} - y_{i+1,j})\Big]$$

Pentru a găsi minimul funcției de mai sus trebuie ca:

$$\frac{\partial E}{\partial y_{i,j}} = 0 : \text{ pentru } \quad 1 \le i \le M; \quad 1 \le j \le N . \tag{4.46}$$

Pe baza condiției de mai sus se poate scrie:

$$\frac{\partial E}{\partial y_{i,j}} = \sum_{i=1}^{M}\sum_{j=1}^{N}\frac{1}{h^2}\Big[-4(y_{i-1,j} - 2y_{i,j} + y_{i+1,j}) - 4(y_{i,j-1} - 2y_{i,j} + y_{i,j+1}) +$$

$$+ y_{i+1,j+1} - y_{i+1,j} - y_{i,j+1} + y_{i,j} + y_{i,j} - y_{i,j-1} - y_{i-1,j} + y_{i-1,j-1} + \quad ,$$

$$+ y_{i-1,j+1} + y_{i,j} - y_{i,j+1} - y_{i-1,j} + y_{i,j} + y_{i+1,j-1} - y_{i,j-1} - y_{i+1,j}\Big]$$

respectiv:

$$\frac{\partial E}{\partial y_{i,j}} = \sum_{i=1}^{M}\sum_{j=1}^{N}\frac{1}{h^2}(20y_{i,j} - 6y_{i-1,j} - 6y_{i+1,j} - 6y_{i,j} - 6y_{i,j+1} + \tag{4.47}$$

$$+ y_{i+1,j+1} + y_{i-1,j-1} + y_{i-1,j+1} + y_{i+1,j-1}) = 0$$

Din egalitatea de mai sus rezultă:

$$y_{i,j} = \frac{6y_{i-1,j} + 6y_{i+1,j} + 6y_{i,j} + 6y_{i,j+1} - (y_{i+1,j+1} + y_{i-1,j-1} + y_{i-1,j+1} + y_{i+1,j-1})}{20} \tag{4.48}$$

Pe de altă parte, pe baza relației (4.16) se poate scrie:

$$\sum_{C_{kl} \in N_r} A_{ij,kl} y_{kl} = a_{-1,-1}y_{i-1,j-1} + a_{-1,+1}y_{i-1,j+1} + a_{+1,+1}y_{i+1,j+1} + a_{+1,-1}y_{i+1,j-1} + \tag{4.48}$$

$$+ a_{-1,0}y_{i-1,j} + a_{+1,0}y_{i+1,j} + a_{0,-1}y_{i,j-1} + a_{0,-1}y_{i,j+1} + a_{0,0}y_{i,j} = y_{i,j}$$

Prin identificarea coeficienților între expresiile (4.47) și (4.48) rezultăelementele operatorului A: (4.49)

$$a_{-1,-1} = a_{-1,+1} = a_{+1,-1} = a_{+1,+1} = -0.05 ,$$

$$a_{-1,0} = a_{0,-1} = a_{0,+1} = a_{+1,0} = 0.3 ,$$

$$a_{0,0} = 0 .$$

Rezultă în final operatorul de reacție aintpol3.tem, care asigură interpolare spline cubică, sub forma: (4.50)

$$A = \begin{bmatrix} -0.05 & 0.3 & -0.05 \\ 0.3 & 0 & 0.3 \\ -0.05 & 0.3 & -0.05 \end{bmatrix} \quad B = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix} \quad z = \begin{bmatrix} 0 \end{bmatrix}$$

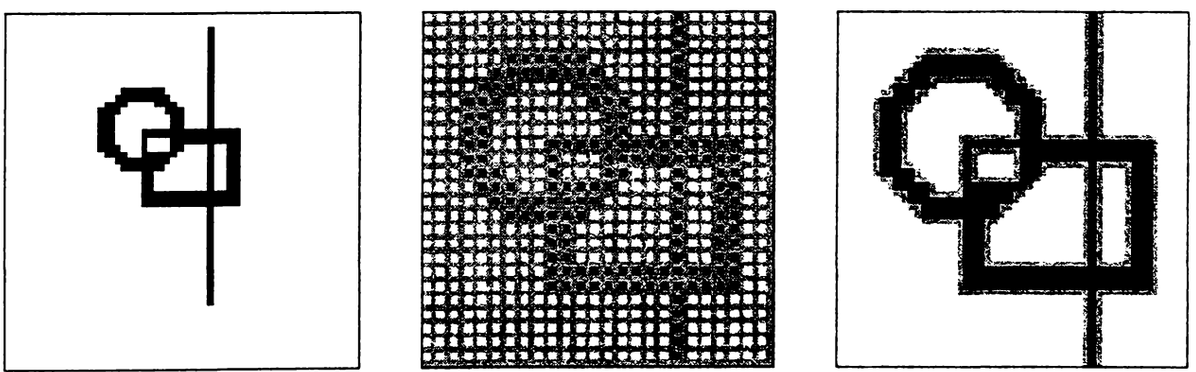
4.4 Testarea metodelor de interpolare CNN prin simulare

Rezultatele experimentale prezentate în continuare au fost obținute cu ajutorul unor programe scrise în limbajul de asamblare "AMC" (Extended analogic macro code and interpreter [130]) (anexa 4) şi executate pe platforma "VisMouse-SimCNN" (Visual mouse platform and multi-layer CNN simulator [124]) din mediul de dezvoltare "CadetWin" (CNN application development environment and toolkit under Windows [123]). Pentru inserția de elemente cu valori nule pe linii şi coloane, în imaginea ce urmează a fi interpolată, s-a utilizat mediul de programare MATLAB [40].

4.4.1 Mărirea imaginilor cu păstrarea dimensiunilor elementului de imagine

4.4.1.1 Mărirea imaginilor binare

În figura 4.3 sunt prezentate imaginile binare utilizate pentru exemplificarea procesului de dublare a dimensiunilor imaginii pe orizontală şi pe verticală, cu păstrarea dimensiunilor elementelor de imagine ca la imaginea inițială (50*50 pixeli). Pentru a se putea evalua eroarea, s-a creat o imagine considerată de referință printr-o așa zisă interpolare ideală (figura 4.3c).



(a) (b) (c)

*Figura 4.3. Imagini binare utilizate pentru testarea metodelor de interpolare cu CNN utilizând operatorii aintpol1.tem, aintpol2.tem, aintpol3.tem: (a) imaginea inițială de 50 *50 de pixeli, din care se decupează o parte; (b) imaginea de stare având 50 *50 de pixeli, la care s-au intercalat pe linii şi coloane elemente cu valori nule; (c) imaginea de referință rezultată în urma unei măriri prin interpolare ideală.*

Erorile aparțin tot domeniului de valori [-1, +1]. Imaginile binare care rezultă în urma interpolării CNN utilizând operatori cu reacție, precum și erorile aferente ale acestora față de imaginea de referință, sunt prezentate în figura 4.4.

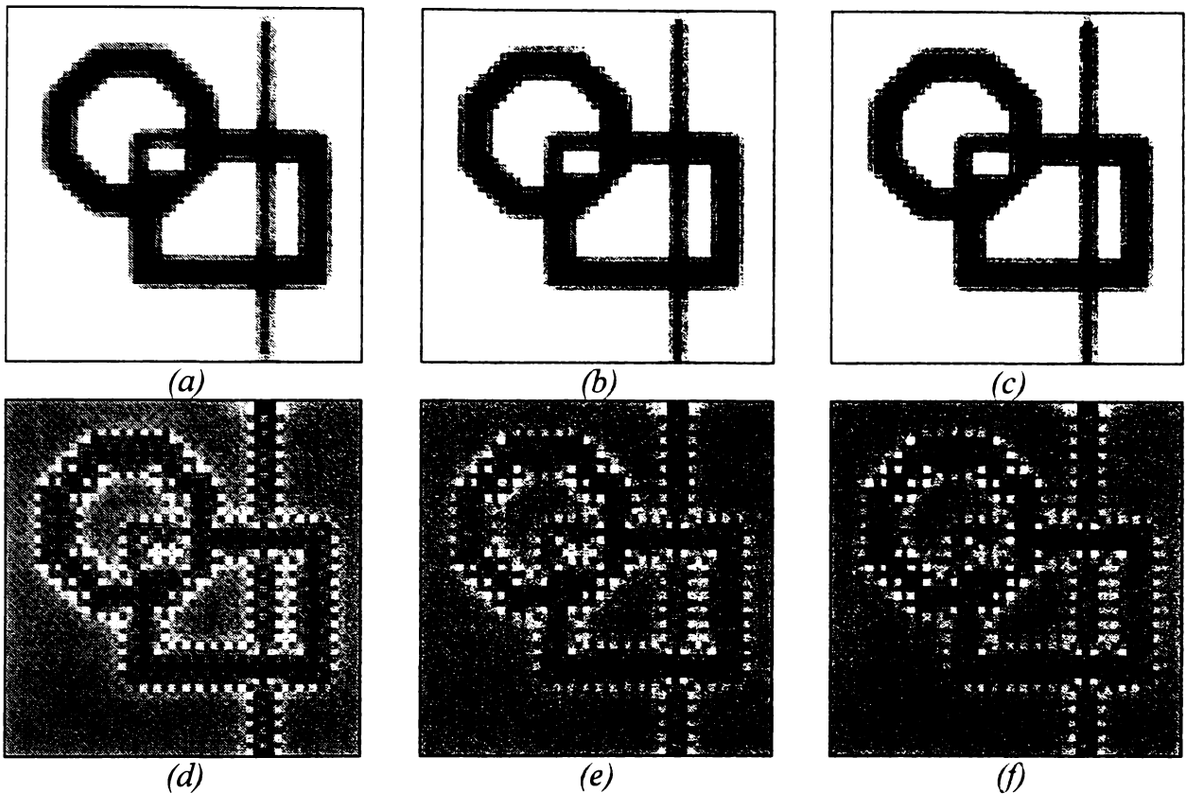


Figura 4.4. Imagini binare obținute în urma interpolării prin simulare: (a) imaginea rezultată prin interpolare cu aintpol1.tem; (b) imaginea rezultată prin interpolare cu aintpol2.tem; (c) imaginea rezultată prin interpolare cu aintpol3.tem; (d) eroarea dintre imaginea de referință și imaginea rezultată prin interpolare cu aintpol1.tem; (e) eroarea dintre imaginea de referință și imaginea rezultată prin interpolare cu aintpol2.tem; (f) eroarea dintre imaginea de referință și imaginea rezultată prin interpolare cu aintpol3.tem.

Timpul de procesare a interpolării CNN utilizând operatori de reacție depinde de corelația existentă între pixelii imaginii, adică de caracterul imaginii și mai puțin de dimensiunea ei. În cazul simulării, adică a implementării numerice CNN, mărimea timpului de procesare rezultă ca fiind numărul minim de iterații necesar pentru obținerea unei imagini stabile. În aplicațiile prezentate anterior, pentru aceste imagini binare timpul de procesare necesar a fost aproximativ 24 τ_{CNN}, τ_{CNN} fiind constanta de timp a unei celule.

4.4.1.2 Creşterea rezoluţiei imaginilor cu niveluri de gri

Alte rezultate privind interpolarea CNN cu operatori de reacţie obţinute prin simulare, sunt prezentate în continuare în figura 4.5.

Dintr-o imagine cu niveluri de gri iniţială s-a selectat o regiune de interes cu dimensiunea de 50*50 pixeli, ca în figura 4.5a. În această imagine s-a inserat pe linii şi pe coloane zerouri, rezultând o imagine cu dimensiunea de 100*100 pixeli. După acesta, prin interpolare cu aintpol3.tem, a rezultat imaginea din figura 4.5b. Din imaginea rezultată s-a decupat o altă zonă de interes cu dimensiunea 50*50 pixeli, urmat de o nouă interpolare rezultând o imagine cu dimensiunea 100*100 pixeli, prezentată în figura 4.5c. Timpul total de procesare la interpolare pentru imaginile cu niveluri de gri de dimensiunea menţionată mai sus, este aproximativ 24 τ_{CNN} .

(a) *(b)* *(c)*

Figura 4.5. Exemple de interpolare CNN a unor imagini utilizând aintpol3.tem: (a) imaginea iniţială; (b) imaginea rezultată după prima etapă de mărire; (c) imaginea rezultată după a doua etapă de mărire.

4.4.2 Reconstrucţia de imagini prin interpolare

4.4.2.1 Reconstrucţia prin interpolare a imaginilor în care elementele de imagine cu valori cunoscute au o dispunere neregulată

În vederea evaluării performanţelor reconstrucţiei imaginilor prin interpolare s-a ales o imagine în care pixelii cu valori cunoscute au o dispunere neregulată, aleatoare. Şi în acest caz rezoluţia şi dimensiunile imaginii rămân nemodificate. Aceste elemente cu valori cunoscute nu-şi vor schimba starea în decursul procesării fiind utilizată în acest scop o imagine mască. Valorile elementelor de imagine care nu se cunosc se vor calcula prin interpolare.

În exemplul anterior elementele de imagine cu valori cunoscute reprezentau 25% din numărul total de pixeli. În acest exemplu s-a scăzut acest procent la numai 15% din numărul total de elemente de imagine. În figura 4.6 se prezintă rezultatele reconstrucției de imagini (100*100 pixeli) prin interpolare CNN, utilizând operatorii aintpol1.tem, aintpol2.tem şi aintpol3.tem.

(a) (b) (c) (d)

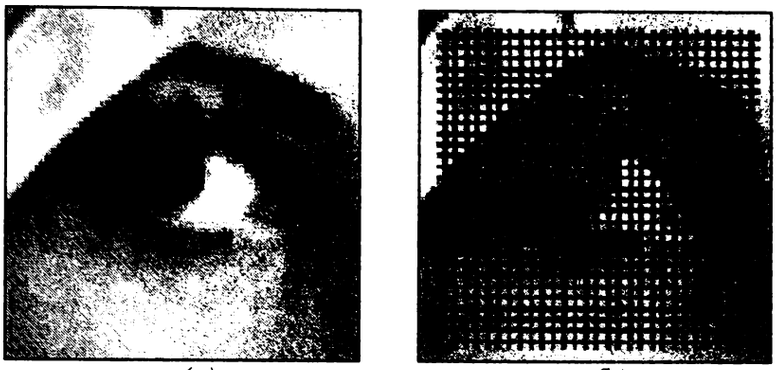
Figura 4.6. Reconstrucția de imagini prin interpolare: (a) imaginea de intrare în care se cunosc numai 15% din totalul de valori ale elementelor de imagine; (b) imaginea rezultată prin interpolare cu aintpol1.tem; (c) imaginea rezultată prin interpolare cu aintpol2.tem; (d) imaginea rezultată prin interpolare cu aintpol3.tem.

La reconstrucția prin interpolare a imaginilor cu niveluri de gri, în care elementele de imagini cu valori cunoscute au o dispunere neregulată, timpul de procesare este în medie de 50 τ_{CNN}, deci creşte față de situația prezentată anterior chiar la aceiaşi dimensiune a imaginii.

4.4.2.2 Reconstrucția prin interpolare a imaginilor în care elementele de imagine cu valori cunoscute au o dispunere regulată

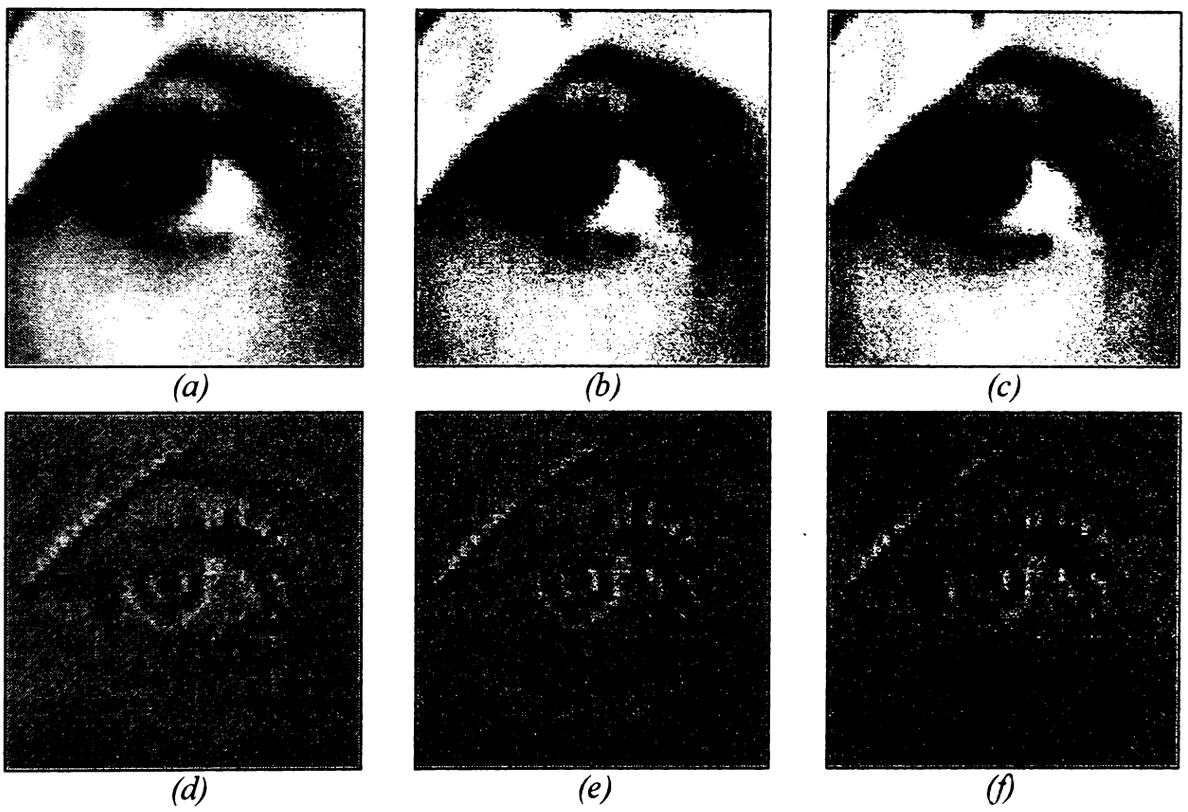
Reconstrucția de imagini în cazul în care elementele de imagine ale căror valori se vor calcula sunt dispuse regulat pe linii şi coloane intercalate între elementele de imagine ale căror valori se cunosc, este util în evaluarea erorilor metodelor de interpolare cu operatorii determinați anterior.

În acest caz rezoluția imaginii de ieşire rămâne identică, la fel şi dimensiunile. Imaginea de referință utilizată la evaluarea erorii este chiar imaginea inițială, reală. Imaginea de interpolat s-a obținut din imaginea inițială reală prin anularea valorilor unor elemente de imagine pe linii şi coloane. În figura 4.7 sunt prezentate imaginile cu niveluri de gri, de 64*64 pixeli, utilizate pentru testarea metodelor de interpolare cu CNN utilizând operatorii aintpol1.tem, aintpol2.tem, aintpol3.tem. Imaginile cu niveluri de gri, care rezultă în urma interpolării prin simulare, sunt prezentate în figura 4.8.



(a) (b)

Figura 4.7. Imagini cu niveluri de gri utilizate pentru testarea metodelor de interpolare cu CNN utilizând operatorii aintpol1.tem, aintpol2.tem, aintpol3.tem: (a) imaginea originală; (b) imaginea care se interpolează.



(a) (b) (c)

(d) (e) (f)

Figura 4.8. Imagini cu niveluri de gri obținute în urma interpolării prin simulare: (a) imaginea rezultată prin interpolare cu aintpol1.tem; (b) imaginea rezultată prin interpolare cu aintpol2.tem; (c) imaginea rezultată prin interpolare cu aintpol3.tem; (d) eroarea dintre imaginea originală şi imaginea rezultată prin interpolare cu aintpol1.tem; (e) eroarea dintre imaginea originală şi imaginea rezultată prin interpolare cu aintpol2.tem; (f) eroarea dintre imaginea originală şi imaginea rezultată prin interpolare cu aintpol3.tem.

La reconstrucția prin interpolare a imaginilor în care elementele de imagine cu valori cunoscute au o dispunere regulată, se constată o reducere substanțială a timpului de procesare a interpolării, în medie până la 3 τ_{CNN}. Acest lucru este uşor de înțeles dacă se are în vedere gradul înalt de corelație care există între pixelii imaginii originale.

4.4.3 Codarea şi decodarea imaginilor prin interpolare

În vederea reducerii volumului de date care se transmite la distanță se poate folosi interpolarea pentru micşorarea dimensiunilor imaginii la emisie respectiv la refacerea dimensiunilor imaginii de la recepție [23].

Având o imagine cu niveluri de gri de dimensiune M*N pixeli (figura 4.9a) pentru codare se recalculează prin interpolare valorile elementelor de imagine care sunt notate cu 1. Pixelii de culoarea albă reprezintă elementele de imagine la care se cunosc valorile inițiale. Imaginea mască corespunzătoare nu permite modificarea valorilor acestor pixeli (celule). Din imaginea rezultată după interpolare se va selecta o imagine de dimensiune M/2*N/2 pixeli, formată numai din elementele cu valori recalculate (notate cu 1) şi care se va transmite pe canal.

La recepție, se refac dimensiunile imaginii inițiale M*N pixeli, prin intercalarea elementelor de imagine recepționate cu elemente de imagini având valori inițiale zero (figura 4.9b). Pentru decodare se foloseşte acelaşi procedeu de interpolare ca şi la codare. În imaginea de interpolat la decodare, valorile elementelor recepționate, simbolizate cu alb, îşi vor menține constant nivelul şi se vor calcula prin interpolare valorile elementelor de imagine notate cu 2, simbolizate cu gri.

La acelaşi număr de biți utilizați pentru codarea valorii un pixel, prin metoda de codare-decodare expusă, se transmite pe canal o imagine de M/2*N/2 pixeli în locul unei imagini cu dimensiunea M*N pixeli, adică se reduce la 25% cantitatea de informație necesară a fi transmisă.

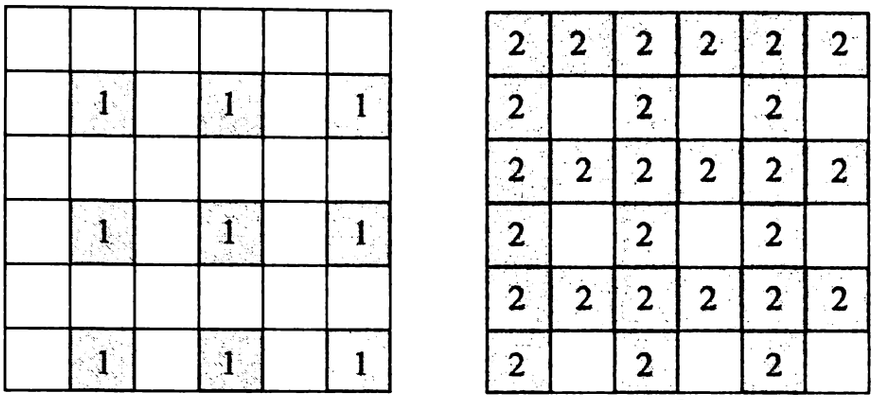


Figura 4.9. Principiul codării-decodării prin interpolare CNN: (a) imaginea de interpolat la codare; (b) imaginea de interpolat la decodare.
Prin interpolare are loc determinarea elementelor de imagine notate cu 1, la codare şi a celor notate cu 2, la decodare.

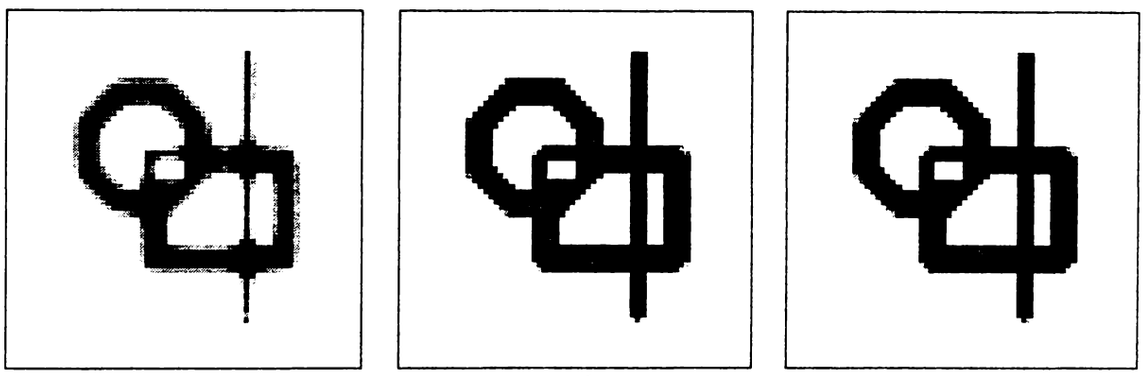
În figura 4.10 se prezintă imaginea inițială de dimensiune 256*256 pixeli înainte codării şi imaginea obținută după decodare cu dimensiunea refăcută de 256*256 pixeli. La codarea şi decodarea imaginii s-a folosit interpolarea CNN cu operatorul aintpol3.tem. Timpul de procesare la decodare este aproximativ 25 τ_{CNN}, la codare acest număr este mult mai mic având în vedere faptul că efectul de propagare între celule este împiedicat de structura măştii.

Figura 4.10. Utilizarea aintpol3.tem pentru codarea şi decodarea imaginilor: (a) imaginea originală înainte de codare; (b) imaginea cu dimensiunea reconstituită, după decodare.

4.5 Implementarea metodelor de interpolare pe un chip-CNN de 64*64 pixeli

Funcționarea operatorilor cu reacție elaborați pentru interpolare a fost testată pe baza unor programe în limbaj de asamblare AMC (anexa 4) și pe un chip-CNN de 64 * 64 pixeli (cP 4000 [68]), utilizând interfața "CCPS" (CNN chip prototyping system [131]) din mediul de dezvoltare "CadetWin".

Imaginile inițiale cu dimensiunea 64*64 pixeli care s-au procesat sunt identice cu imaginile binare și cu niveluri de gri de la simulare. Rezultatele obținute la interpolarea imaginilor binare și cu niveluri de gri la testarea pe chip a operatorilor aintpol1.tem, aintpol2.tem, aintpol3.tem sunt prezentate în figura 4.11 și figura 4.12.



(a) *(b)* *(c)*

*Figura 4.11. Imagini binare obținute prin interpolare utilizând un chip de 64*64 pixeli: (a) imaginea rezultată prin interpolare cu aintpol1.tem; (b) imaginea rezultată prin interpolare cu aintpol2.tem; (c) imaginea rezultată prin interpolare cu aintpol3.tem.*

Interpolarea imaginilor binare și cu niveluri de gri, executate pe chip cu ajutorul operatorilor aintpol1.tem, aintpol2.tem, aintpol3.tem, au necesitat în medie 3,4 ms.

În aceste exemple numărul total de iterații necesar până la obținerea imaginii finale este dependent și de operatorul utilizat. La aintpol1.tem timpii de procesare au fost 10 τ_{CNN}, dar pentru aintpol2.tem și aintpol3.tem erau necesare 25 τ_{CNN}.

Apreciind imaginile binare și cu niveluri de gri obținute prin interpolare, atât în cazul simulării cât și în cazul testării pe chip, se constată că rezultate mai bune se obțin cu operatorii aintpol3.tem și aintpol2.tem. În cazul imaginilor cu niveluri de gri rezultate prin interpolare pe chip, se poate constata mai ușor decât la simulare,

comportamentul mai bun al operatorului aintpol3.tem din punct de vedere al contrastului.

(a) *(b)* *(c)*

*Figura 4.12. Imagini obținute prin interpolăre utilizând chipul de 64*64 pixeli: (a) imaginea rezultată prin interpolare cu aintpol1.tem; (b) imaginea rezultată prin interpolare cu aintpol2.tem; (c) imaginea rezultată prin interpolare cu aintpol3.tem.*

În tabelul 4.1 se prezintă o situație comparativă privind timpul mediu de procesare aferent celor 3 operatori aintpol1.tem, aintpol2.tem, aintpol3.tem, în cazul simulării respectiv la procesarea pe chip. Au fost avute în vedere cazul măririi de imagini binare respectiv reconstrucția prin interpolare a imaginilor cu niveluri de gri, în care elementele de imagine cu valori cunoscute au o dispunere regulată. Timpul mediu complet de procesare rezultă în urma efectuării complete a programului AMC de interpolare a unei imagini, la simulare respectiv la implementare pe un chip-CNN. Acest timp include alături de timpul efectiv de procesare la interpolare şi timpii necesari pentru citirea imaginii din memorie respectiv pentru încărcarea şi descărcarea imaginii pe chip precum şi timpul necesar pentru afişarea imaginii.

Operator	Simulare PC 300 MHz		Procesare pe chip-CNN 64*64 pixeli
	Timpul mediu de procesare în τ_{CNN}	Timpul mediu complet de procesare	Timpul mediu complet de procesare
aintpol1.tem	24	0.75s	3.4ms
aintpol2.tem	24	0.75s	3.4ms
aintpol3.tem	24	0.75s	3.4ms

Tabel 4.1. Caracteristicile interpolării la simulare şi procesare pe chip.

De remarcat faptul că deocamdată din timpul total de procesare pe chip o valoare semnificativă din acesta este utilizat pentru încărcarea şi descărcarea imaginii pe chip. Dacă în cazul unui chip-CNN s-ar considera numai timpul efectiv de procesare de la interpolare, pentru o constantă τ_{CNN} ce caracterizează o celulă de 250 ns, ar rezulta că prelucrarea analogică pe chip a unei imagini cu dimensiunea de 100*100 pixeli nu ar dura mai mult de 6,25μs (25 iterații*250 ns). Acest timp este deja comparabil cu durata impulsului de stingere pe orizontală în cazul semnalului analogic video complex standard. Astfel, după fiecare linie de imagine recepționată, chiar pe durata impulsului de stingere pe orizontală poate fi realizată câte o interpolare CNN.

4.6 Concluzii

În acest capitol s-au prezentat modalități noi, originale de interpolare a semnalelor bidimensionale cu rețele neuronale celulare. Interpolarea imaginilor binare şi cu niveluri de gri cu ajutorul rețelele neuronale celulare care utilizează numai operatorii de reacție liniari de dimensiune 3*3 se poate implementa pe un chip-CNN [68]. Față de procedeele de interpolare cu rețele neuronale celulare existente, interpolarea cu operatorii propuşi de autor se poate efectua în timp real, prin reducerea substanțială a numărului de încărcări pe chip a imaginilor inițiale. Datorită procesării complet paralele, timpul de execuție nu se modifică proporțional odată cu creşterea dimensiunilor imaginii.

Utilizând programe scrise în limbaj de asamblare AMC, cu operatorii determinați s-a verificat prin simulare şi testare pe un chip-CNN funcționarea corectă a acestor metode de interpolare CNN.

Cu ajutorul operatorilor propuşi în acest capitol, pe baza rezultatelor obținute la testare, interpolarea semnalelor bidimensionale cu rețele neuronale celulare poate constitui o metodă utilă la mărirea şi interpretarea zonelor de interes major din imaginile biomedicale şi la obținerea de imagini de calitate HDTV la recepționarea semnalului TV analogic uzual, cu păstrarea dimensiunilor elementului de imagine original.

Având în vedere viteza de procesare ridicată metodele propuse de interpolare pot servi la obținerea imaginilor de comandă ale unui robot, când deja se cunosc valorile unor elemente de imagine corespunzătoare pentru anumite puncte de referință din spațiu.

Interpolarea CNN în timp real cu operatorii propuşi, poate constitui o modalitate eficientă şi rapidă de reconstrucție a unor imagini deteriorate ori cunoscute parțial sau la codarea şi decodarea imaginilor în vederea transmiterii a acestora.

Respectând condițiile inițiale specificate, metoda propusă în acest capitol, pentru determinarea pe cale analitică a operatorilor CNN, poate fi utilă la şi în alte aplicații în care se utilizează imagini cu niveluri de gri, dacă se poate identifica o funcție de cost potrivită.

5. Planificarea traiectoriei unui robot mobil într-un mediu cu obstacole utilizând rețele neuronale celulare

În acest capitol sunt prezentate metode noi, originale, de planificare bazată pe imagini a traiectoriei unui robot mobil într-un mediu cu obstacole [54]. Pentru prelucrarea imaginilor şi planificarea traiectoriei unui robot se utilizează rețele neuronale celulare, deoarece prin implementarea acestei metode pe chip-CNN se poate asigura procesarea în timp real a imaginilor [7,8,68,87]. Între poziția de start şi poziția țintă, ocolind obstacolele din mediu, algoritmul propus furnizează robotului traiectoria optimă din punct de vedere al lungimii şi a numărului de viraje.

5.1 Planificarea traiectoriei unui robot mobil într-un mediu cu obstacole

O problemă importantă în robotică este planificarea traiectoriei unui robot mobil într-un mediu cu obstacole, pornind de la o poziție inițială de start până se ajunge la o poziție țintă. Pentru a rezolva această problemă, în literatura de specialitate există o multitudine de soluții care în general au la bază două metode [25,41,52,65]:

i.) Planificarea traiectoriei sau navigarea robotului prin metodă globală.

Această metodă este asociată cu nivelul ierarhic superior din sistemul de conducerea al unui robot. Cu această metodă se determină o traiectorie optimă, ocolindu-se obstacolele statice sau mobile cunoscute din mediu. De cele mai multe ori metodele globale au la bază crearea de "hărți" şi abordează problema numai din punct de vedere geometric.

ii.) Planificarea traiectoriei sau navigarea robotului prin metodă locală.

Această metodă se asociază cu nivelul ierarhic inferior din sistemul de conducere al unui robot şi de aceea realizează conducerea robotului pe o traiectorie prescrisă de nivelul de planificare globală. Folosind metoda locală de navigare pentru un robot, se pot evita obstacolele necunoscute, statice sau în mişcare, compensând incertitudinea datelor furnizate de nivelul global. Planificarea sau navigarea locală poate lua în

considerare atât cinematica cât şi dinamica robotului, deoarece se bazează pe informaţii obţinute din semnale, care se pot prelucra în timp real [25,72]. Însă prin folosirea numai a informaţiilor locale nu se garantează găsirea soluţiei optime pentru traiectoria planificată robotului şi nu se semnalează dacă poziţia ţintă nu este accesibilă.

Deseori se combină cele două strategii, adică planificarea globală pentru asigurarea traiectoriei accesibile şi navigaţia locală pentru optimizarea locală a traiectoriei şi evitarea obstacolelor neaşteptate. Pentru navigaţia locală se poate utiliza metoda grafurilor şi metoda câmpurilor de potenţial.

Pentru determinarea traiectoriei cu ajutorul grafurilor, mediul de lucru este reprezentat printr-un graf în care nodurile reprezintă poziţiile accesibile [52]. Două noduri sunt conectate între ele numai dacă nu există obstacole între ele. Pentru traseul dintre cele două noduri se asociază o anumită pondere corespunzătoare distanţei dintre noduri.

În cazul metodei câmpurilor de potenţial mediul de lucru este reprezentat printr-un set de puncte libere, unde nu există obstacol. În fiecare punct valoarea potenţialului se exprimă prin funcţia [56,94]:

$$\Phi(p) = f(d_i(g), d_1(ob_1), ..., d_n(ob_n)), \qquad (5.1)$$

unde $d_i(g)$ este distanţa dintre poziţia actuală p şi poziţia ţintă, iar $d_n(ob_n)$ este distanţa dintre punct şi un obstacol. Din poziţia actuală, robotul se va deplasa în următoarea poziţie care are potenţialul cel mai scăzut.

Reţelele neuronale, în general, se pot utiliza pentru navigarea unui robot mobil într-un mediu cu obstacole [41,72,82]. Utilizând reţelele neuronale celulare, mediul este discretizat într-o imagine şi astfel reprezentat printr-o reţea CNN standard de M*N celule [94]. Metoda propusă se bazează pe efectul de atracţie de către celula ţintă asupra celulelor din vecinătatea sa cu o forţă F. La rândul lor şi aceste celule vor exercita o forţă asupra celulelor din vecinătatea lor, ş.a.m.d. până când se ajunge la celula care reprezintă poziţia curentă a robotului. Mişcarea robotului se va realiza pe direcţia forţei de atracţie maxime. Această metodă semnalează dacă nu există nici o traiectorie liberă de la poziţia actuală a robotului la poziţia ţintă, însă se poate implementa numai pe o reţea neuronală celulară cu mai multe straturi.

Cu ajutorul reţelelor neuronale celulare se poate realiza conducerea unui robot mobil pe baza informaţiei vizuale, pe o traiectorie marcată, prin compararea imaginilor achiziţionate curente cu imagini de referinţă memorate [99].

Rețelele neuronale celulare se pot utiliza la planificarea traiectoriei, pentru conducerea mai multor roboți mobili într-un mediu de lucru cu obstacole, fără a se produce coliziune [56].

Planificarea traiectoriei unui robot într-un mediu cu obstacole se poate realiza şi prin utilizarea automatului celular bidimensional [106] implementat VLSI.

Algoritmul CNN de proiectare a cablajelor imprimate [109], care utilizează o metodă de detectare a drumului cel mai scurt între două puncte este o aplicație asemănătoare problemei planificării traiectoriei prescrise pentru un robot mobil într-un mediu cu obstacole. Determinarea traseului este posibilă numai respectând anumite condiții inițiale impuse pentru imaginea cablajului.

În cadrul algoritmului CNN de planificarea traiectoriei unui robot mobil într-un mediu cu obstacole, propus de autor în acest capitol, pentru prelucrarea imaginilor se foloseşte numai rețea neuronală celulară invariantă în spațiu, cu operatori de dimensiune 3*3. În imaginile binare, valorile pixelilor sunt +1 pentru negru şi –1 pentru alb, iar pentru imaginile cu niveluri de gri valorile pixelilor sunt în domeniul standard CNN, adică [-1, +1], de la alb către negru.

5.2 Algoritmul CNN de comandă a unui robot mobil cu reacție vizuală bazată pe imagini

În aplicația de față, se consideră un robot mobil care se găsește într-un mediu plan cu obstacole statice (figura 5.1), supravegherea lui fiind realizată cu o singură cameră video. Imaginile reale ale mediului sunt achiziționate la momente discrete de timp (t_0+kT), începând de la momentul inițial t_0, cu pasul de eşantionare T, k fiind un număr natural, k∈**N**. Comanda vizuală a robotului se realizează pe baza algoritmului global prezentat în figura 5.2. Între două eşantioane de imagini succesiv achiziționate, pentru o imagine de intrare curentă trebuie să se realizeze toate prelucrările necesare din algoritm. De aceea, achiziția unei noi imagini a mediului este posibilă numai după efectuarea tuturor paşilor de procesare din algoritm. Dacă în mediul de lucru nu sunt numai obiecte statice ci şi obiecte în mişcare, adică structura mediului se modifică, este cu atât mai important ca procesarea imaginilor să se facă în timp real. Astfel, dacă comanda vizuală a robotului se efectuează ciclic, pe baza algoritmului global, poziția țintă se poate considera fixă iar poziția de start este chiar poziția actuală a robotului din ultima imagine care s-a achiziționat. Se poate accepta şi situația în care şi obiectul țintă îşi poate modifica poziția de la un cadru la altul a imaginii achiziționate, dar numai dacă viteza de prelucrare a imaginilor este suficient de mare şi astfel, este posibilă obținerea în timp real a mărimii de comandă a robotului.

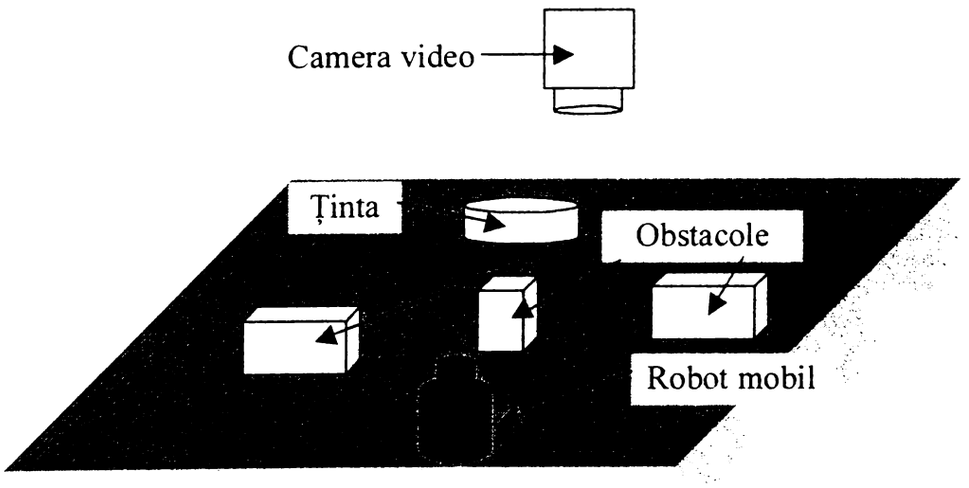


Figura 5.1. Mediul de lucru al unui robot mobil cu comandă vizuală bazată pe imagini.

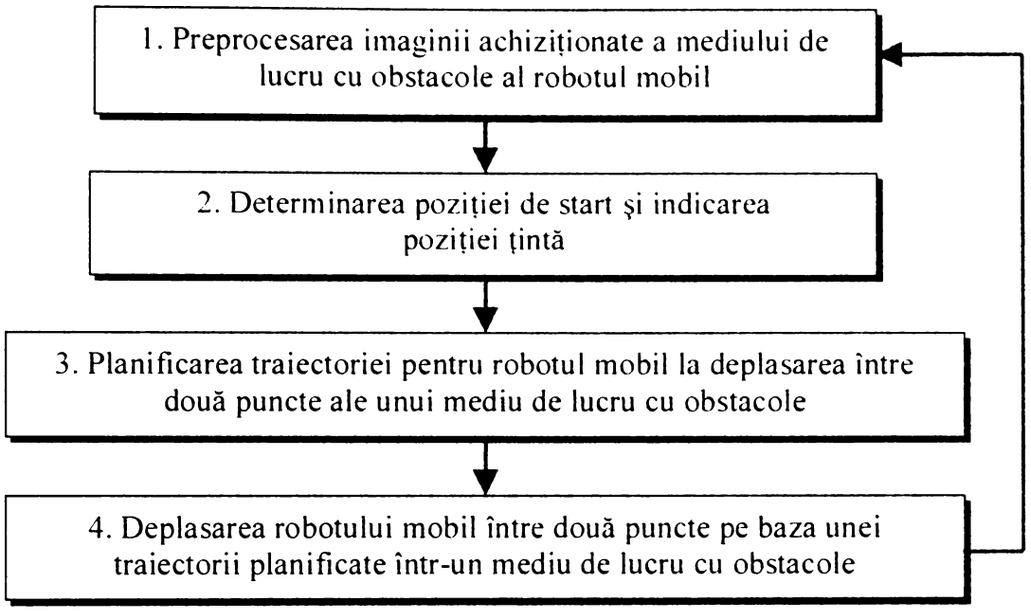


Figura 5.2. Organigrama algoritmului global de comandă vizuală al robotului mobil pentru deplasarea între două puncte ale unui mediu de lucru cu obstacole.

5.3 Planificarea traiectoriei unui robot mobil la deplasarea între două puncte ale unui mediu cu obstacole

Algoritmul propus pentru planificarea traiectoriei unui robot mobil, bazată pe imagini, este prezentat în figura 5.3.

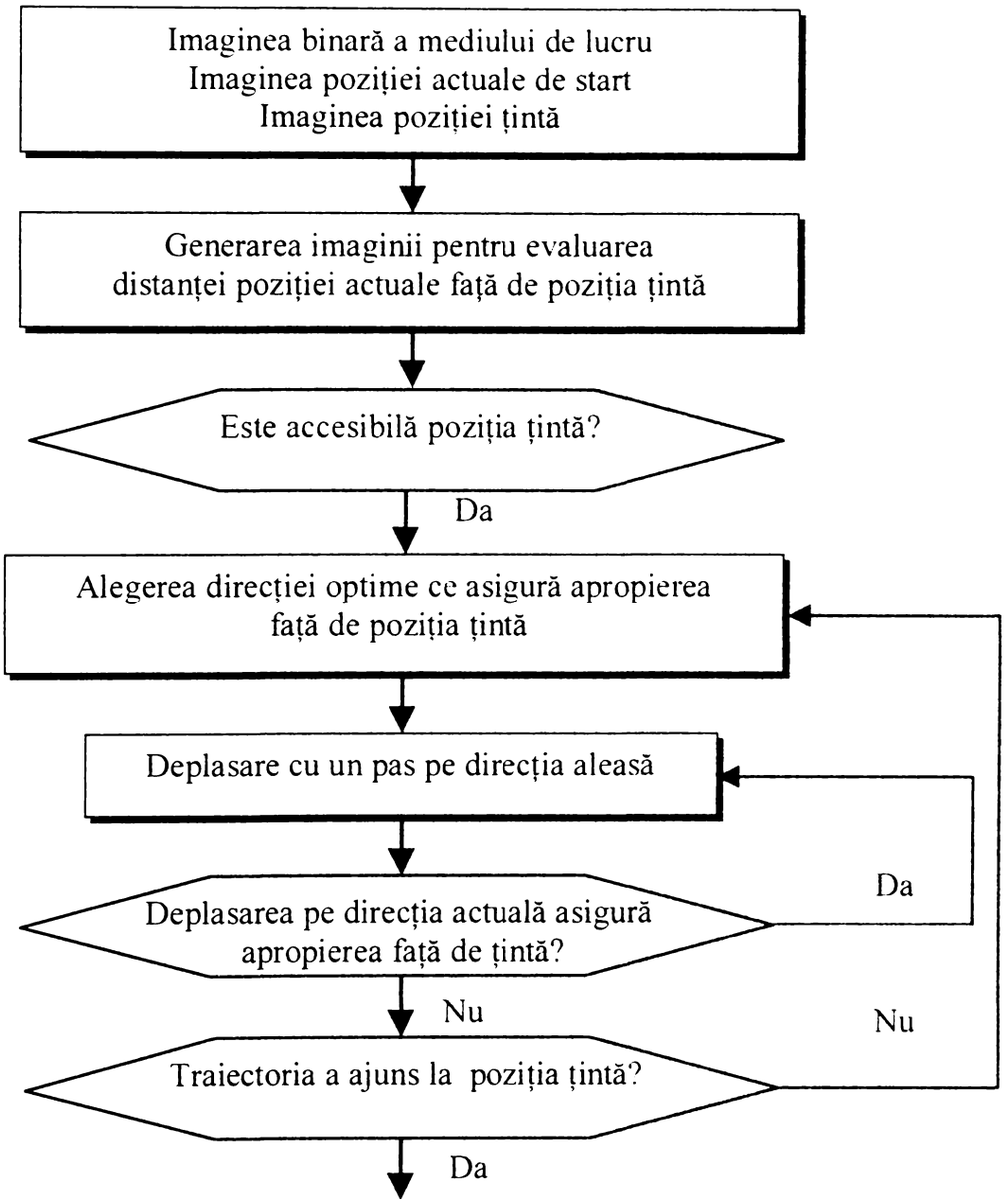


Figura 5.3. Organigrama algoritmului pentru planificarea traiectoriei robotului mobil la deplasarea între două puncte într-un mediu de lucru cu obstacole.

În acest algoritm se utilizează pentru prelucrarea imaginilor numai rețele neuronale celulare. Imaginea de intrare binară a mediului de lucru se obține în urma unor procesări elementare CNN asupra imaginii reale achiziționate cu niveluri de gri. În

imaginea binară rezultată, pentru pixelii cu valori +1 se asociază poziții interzise care nu sunt accesibile robotului în mediul de lucru, iar pentru pixelii cu valoarea –1 se asociază poziții libere, accesibile pentru robot.

5.3.1 Evaluarea distanțelor punctelor din mediul de lucru față de punctul țintă

Pentru evaluarea distanțelor punctelor din mediul de lucru față de punctul țintă, în această etapă se generează o undă în planul imaginii cu centrul sursei situate chiar în punctul țintă. În domeniul aplicațiilor CNN generarea controlată a unei unde este o metodă eficientă care se poate utiliza la detectarea contururilor [86] sau chiar la clasificarea binară [34].

Pentru obținerea imaginii cu evaluarea distanțelor pozițiilor din spațiul de lucru față de poziția țintă se utilizează operatorul explore.tem [109,125], definit de relațiile:

(5.2)

$$A = \begin{bmatrix} 0 & a & 0 \\ a & 1 & a \\ 0 & a & 0 \end{bmatrix} \qquad B = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix} \qquad z = \boxed{0}$$

Reprezentarea grafică a funcției este prezentată în figura 5.4, β având semnificația de unitate de măsură a distanței.

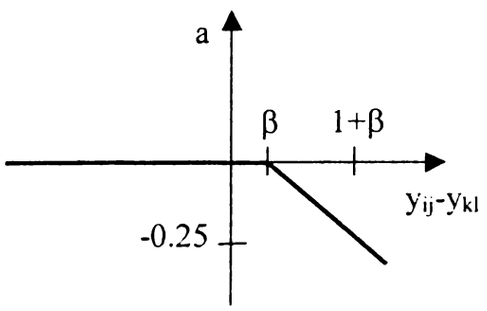


Figura 5.4. Caracteristica de transfer a.

Prin propagarea ei, unda explorează toate căile accesibile ale mediului de lucru începând din punctul țintă (figura 5.5). Pe starea inițială a rețelei $x(t_0)$, STATE, se aplică o imagine în care toate elementele imaginii au valoarea +1, în afară de pixelul corespunzător punctului țintă, care are valoarea –1. Această imagine este de fapt inversa imaginii cu care se indică poziția țintă. Imaginea binară curentă a mediului se va utiliza ca imagine mască (MASK) pentru această prelucrare. Pixelii de la marginea imaginii

mediului de lucru, respectiv frontierei rețelei neuronale celulare, întotdeauna sunt considerați poziții interzise.

Ca urmare a acestei prelucrări în imaginea de ieşire, pixelul din poziția țintă va rămâne la valoarea inițială –1, iar elementele de imagine care vor avea valoarea +1 vor constitui poziții interzise sau inaccesibile robotului. Toți ceilalți pixeli vor avea valori proporționale cu distanțele pozițiilor lor față de poziția țintă. Astfel, în imaginea rezultată, pornind din centrul sursei undei, valoarea pixelului creşte aproximativ cu câte o unitate de măsură a distanței β, la creşterea cu o unitate a razei undei cu propagare circulară.

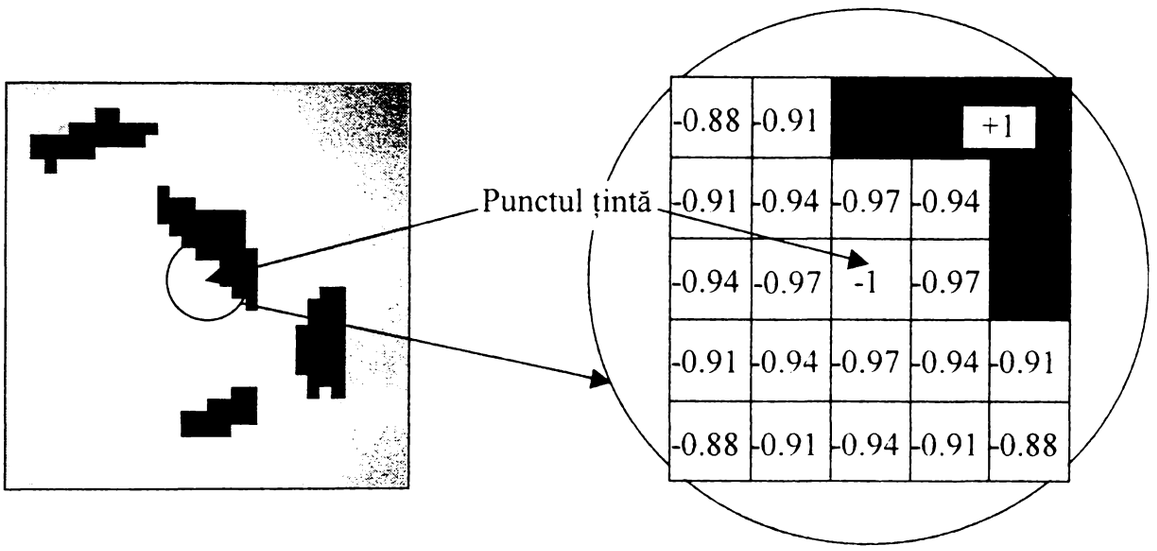


Figura 5.5. Principiul determinării distanței față de țintă prin propagarea unei unde din punctul țintă. Valorile pixelilor cresc cvasiproporțional cu distanța față de punctul origine a sursei.

Dacă există cel puțin o posibilitate de a se ajunge de la poziția de start la poziția țintă, atunci în imaginea de ieşire valoarea pixelului din poziția corespunzătoare poziției de start se va modifica de la +1 la o valoare care este proporțională cu distanța dintre poziția de start şi poziția țintă. La fiecare ciclu nou al algoritmului global, propus în acest capitol, se pune în evidență dacă nu există nici un drum liber de obstacole de la poziția de start la poziția țintă. Acest lucru este posibil având în vedere că dacă ținta nu este accesibilă valoarea pixelului din poziția start rămâne +1 şi după procesarea CNN, utilizată pentru evaluarea distanțelor punctelor din mediul de lucru față de punctul țintă.

5.3.2 Alegerea direcției optime de deplasare pentru robot

Alegerea pentru robot a direcției optime din punct de vedere al lungimii traiectoriei planificate este o etapă de prelucrare a algoritmului care are la bază posibilitatea extragerii valorii unui pixel cu nivel de gri, dacă poziția sa este precizată printr-un singur pixel (de exemplu, de valoare –1) într-o altă imagine binară mască cu dimensiunile identice ca şi prima imagine, principiu descris la paragraful 3.3.3. De aceea, pentru această etapă de prelucrare este necesară imaginea cu niveluri de gri care conține estimarea distanțelor pozițiilor din mediul de lucru față de poziția țintă şi imaginea binară în care toți pixelii au valoarea +1, în afară de un pixel (-1) prin care se indică poziția punctului de start, ori poziția actuală de alegere a unei direcții noi. Această imagine binară este de fapt inversa imaginii prin care se indică poziția de start. Dintre cei opt vecini care corespund direcțiilor N, S, E, V, S-E, N-E, N-V, S-V, se evaluează şi se alege cu o metodă locală acea celulă vecină cu valoare cea mai mică, care asigură ca prin direcția corespunzătoare, drumul parcurs de robot la țintă să fie cel mai scurt. Pentru alegerea direcției optime se utilizează procesări elementare AMC pentru imagini, având la bază familia de operatori shift.tem care corespund celor opt direcții [130] definiți de relațiile următoare: (5.3)

shifte.tem

$$A = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix} \quad B = \begin{bmatrix} 0 & 0 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix} \quad z = \boxed{0}$$

shiftes.tem

$$A = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix} \quad B = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix} \quad z = \boxed{0}$$

shifts.tem

$$A = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix} \quad B = \begin{bmatrix} 0 & 1 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix} \quad z = \boxed{0}$$

shiftvs.tem

$$A = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix} \quad B = \begin{bmatrix} 0 & 0 & 1 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix} \quad z = \boxed{0}$$

shiftv.tem

A = | 0 | 0 | 0 |
| 0 | 0 | 0 |
| 0 | 0 | 0 |

B = | 0 | 0 | 0 |
| 0 | 0 | 1 |
| 0 | 0 | 0 |

z = | 0 |

shiftvn.tem

A = | 0 | 0 | 0 |
| 0 | 0 | 0 |
| 0 | 0 | 0 |

B = | 0 | 0 | 0 |
| 0 | 0 | 0 |
| 0 | 0 | 1 |

z = | 0 |

shiftn.tem

A = | 0 | 0 | 0 |
| 0 | 0 | 0 |
| 0 | 0 | 0 |

B = | 0 | 0 | 0 |
| 0 | 0 | 0 |
| 0 | 1 | 0 |

z = | 0 |

shiften.tem

A = | 0 | 0 | 0 |
| 0 | 0 | 0 |
| 0 | 0 | 0 |

B = | 0 | 0 | 0 |
| 0 | 0 | 0 |
| 1 | 0 | 0 |

z = | 0 |

De exemplu, în figura 5.6, pentru deplasarea robotului din punctul A de start către țintă, direcția optimă din punct de vedere al lungimi este către punctul B, iar din punctul de viraj D direcția potrivită este către punctul E.

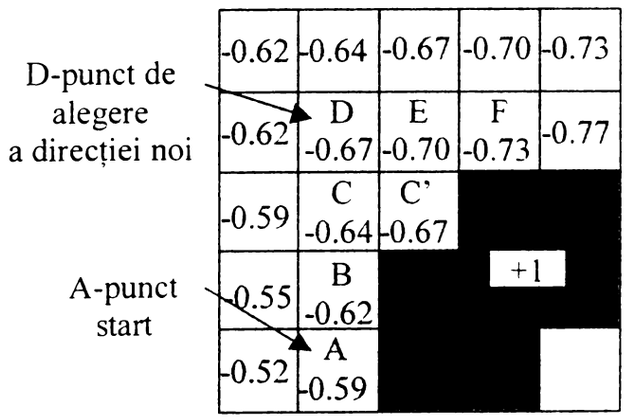


Figura 5.6. Alegerea în punctele de viraj a direcției optime din punct de vedere al lungimii traiectoriei planificate pentru robotul mobil.

5.3.3 Deplasarea continuă a robotului pe o direcția aleasă până se asigură apropierea față de țintă

Deplasarea pe direcția aleasă rămâne neschimbată până când această direcție curentă asigură apropierea robotului de poziția țintă. Aceasta înseamnă că, de pe poziția curentă se va păşi pe următoarea poziție de pe aceeaşi direcție numai dacă valoarea pixelului corespunzător pentru poziția respectivă este mai mică decât valoarea pixelului din poziția curentă, spre exemplu în figura 5.6 punctele B şi C, respectiv C şi D. Dacă valoarea pixelului care urmează pe direcția curentă este mai mare rezultă din nou un viraj sau un pas de alegere a unei noi direcții, punctul D (figura 5.6).

La fiecare ciclu, când se alege o nouă direcție, mai întâi se verifică dacă nu s-a ajuns la poziția țintă, uşor de identificat având în vedere că numai valoarea acestui pixel este –1.

Pentru realizarea imaginii corespunzătoare traiectoriei planificate, după ce s-a ales direcția optimă într-un punct de viraj, pentru apropierea robotului față de poziția țintă, se utilizează familia de operatori select.tem extinsă pe cele opt direcții [109,125], definite de relațiile următoare: (5.4)

selecte.tem

$$A=\begin{bmatrix}0&0&0\\1&4&0\\0&0&0\end{bmatrix}\quad B=\begin{bmatrix}0&0&0\\b&0&0\\0&0&0\end{bmatrix}\quad z=\boxed{1}$$

selectes.tem

$$A=\begin{bmatrix}1&0&0\\0&4&0\\0&0&0\end{bmatrix}\quad B=\begin{bmatrix}b&0&0\\0&0&0\\0&0&0\end{bmatrix}\quad z=\boxed{1}$$

selects.tem

$$A=\begin{bmatrix}0&1&0\\0&4&0\\0&0&0\end{bmatrix}\quad B=\begin{bmatrix}0&b&0\\0&0&0\\0&0&0\end{bmatrix}\quad z=\boxed{1}$$

selectvs.tem

$$A=\begin{bmatrix}0&0&1\\0&4&0\\0&0&0\end{bmatrix}\quad B=\begin{bmatrix}0&0&b\\0&0&0\\0&0&0\end{bmatrix}\quad z=\boxed{1}$$

selectv.tem

A =	0	0	0	B =	0	0	0	z =	1
	0	4	1		0	0	b		
	0	0	0		0	0	0		

selectvn.tem

A =	0	0	0	B =	0	0	0	z =	1
	0	4	0		0	0	0		
	0	0	1		0	0	b		

selectn.tem

A =	0	0	0	B =	0	0	0	z =	1
	0	4	0		0	0	0		
	0	1	0		0	b	0		

selecten.tem

A =	0	0	0	B =	0	0	0	z =	1
	0	4	0		0	0	0		
	1	0	0		b	0	0		

Reprezentarea grafică a funcției neliniare b este prezentată în figura 5.7. β este identică cu cea din figura 5.4, având aceiaşi semnificație de unitate de măsură a distanței în imagine.

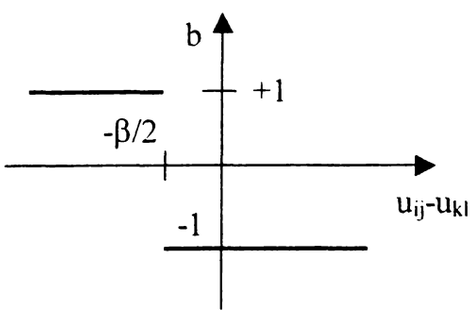


Figura 5.7. Caracteristica de transfer b.

Între poziția de start şi poziția țintă, traiectoria planificată totală pentru robot rezultă din compunerea porțiunilor de traiectorii dintre două viraje care se obțin cu operatorii select.tem (figura 5.8). Pe intrarea INPUT a rețelei neuronale celulare se aplică imaginea cu estimarea distanțelor punctelor mediului de lucru față de poziția țintă. Pe starea STATE se aplică imaginea, cu un singur pixel activ de valoare +1, care indică poziția de start pentru porțiunea din traiectoria planificată. Cei opt operatori select.tem corespund direcțiilor dintre care s-a ales cea optimă în etapa precedentă. De

exemplu, aşa cum rezultă din figura 5.8e, dacă poziţia de start pentru porţiunea de traiectorie curentă este punctul A, utilizând selectn.tem, se realizează porţiunea A-D din traiectoria planificată. În figura 5.8e, vârful săgeţilor indică câte un viraj, adică poziţia în care se alege din nou direcţia care va asigura pentru robotul mobil traiectoria planificată optimă din punctul de vedere al lungimii. Deplasarea robotului pe direcţia aleasă într-un punct de viraj, până ce se asigură apropierea acestuia faţă de poziţia ţintă este o metodă prin care numărul de viraje este minim, ca şi timpul necesar pentru planificarea traiectoriei.

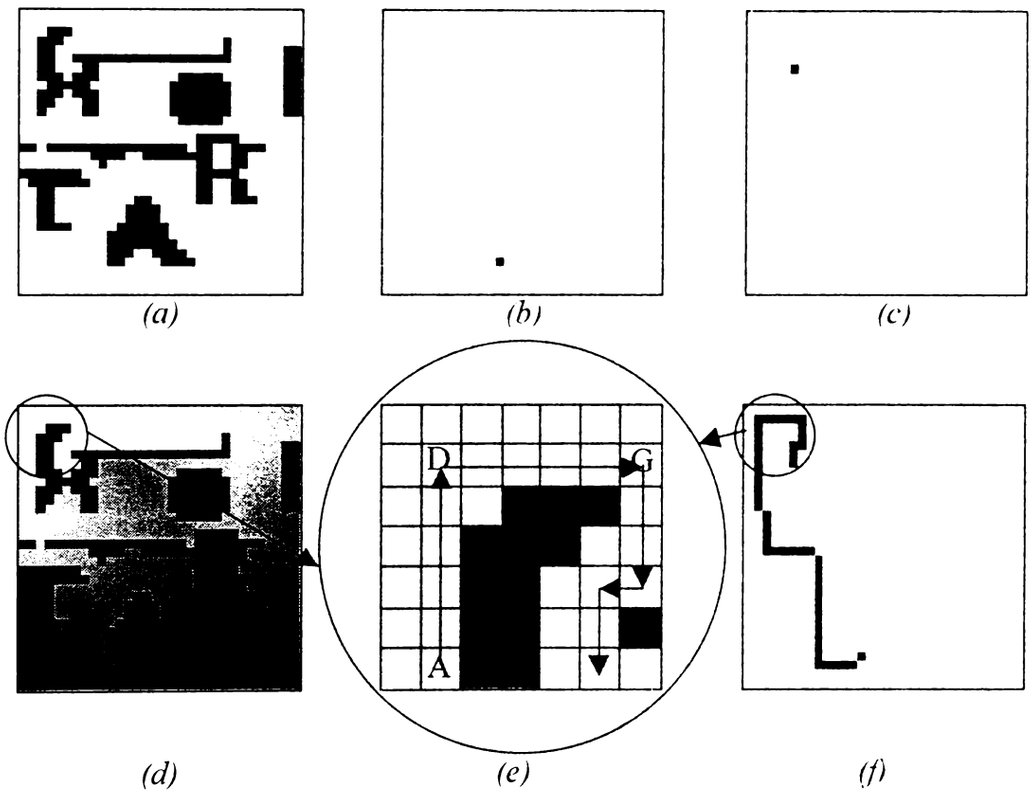


Figura 5.8. Rezultatele simulării algoritmului CNN de planificare a traiectoriei unui robot mobil pentru deplasarea între două puncte ale unui mediu cu obstacole, pe baza principiului efectuării mişcării pe o direcţie aleasă până se realizează apropierea de ţintă: (a) imaginea mediului după preprocesare; (b) determinarea poziţiei de start; (c) indicarea poziţiei ţintă;(d) imaginea generată pentru estimarea distanţei dintre poziţia start şi poziţia ţintă; (e) imaginea în detaliu a unei porţiuni din traiectoria planificată; (f) traiectoria planificată rezultată.

Traiectoria planificată trebuie să rămână de dimensiunea unui pixel după fiecare viraj şi utilizare de operator select.tem. De aceea, imaginea de start, pentru fiecare porţiune de traiectorie, trebuie să conţină un singur pixel activ (cu valoare +1). Acest pixel rezultă ca fiind ultimul element din porţiunea precedentă de traiectorie adică,

ultimul punct de viraj. Pentru îndeplinirea acestei condiții se utilizează și familia de operatori del.tem, extinsă pe cele opt direcții, care s-au determinat pe cale empirică, de către autor. Operatorii del.tem, sunt definiți de relațiile următoare: (5.5)

dele.tem

$$A = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix} \quad B = \begin{bmatrix} 0 & 0 & 0 \\ 1 & 1 & -1 \\ 0 & 0 & 0 \end{bmatrix} \quad z = \boxed{0}$$

deles.tem

$$A = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix} \quad B = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & -1 \end{bmatrix} \quad z = \boxed{0}$$

dels.tem

$$A = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix} \quad B = \begin{bmatrix} 0 & 1 & 0 \\ 0 & 1 & 0 \\ 0 & -1 & 0 \end{bmatrix} \quad z = \boxed{0}$$

delvs.tem

$$A = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix} \quad B = \begin{bmatrix} 0 & 0 & 1 \\ 0 & 1 & 0 \\ -1 & 0 & 0 \end{bmatrix} \quad z = \boxed{0}$$

delv.tem

$$A = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix} \quad B = \begin{bmatrix} 0 & 0 & 0 \\ -1 & 1 & 1 \\ 0 & 0 & 0 \end{bmatrix} \quad z = \boxed{0}$$

delvn.tem

$$A = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix} \quad B = \begin{bmatrix} -1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \quad z = \boxed{0}$$

deln.tem

$$A = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix} \quad B = \begin{bmatrix} 0 & -1 & 0 \\ 0 & 1 & 0 \\ 0 & 1 & 0 \end{bmatrix} \quad z = \boxed{0}$$

delen.tem

$$A = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix} \quad B = \begin{bmatrix} 0 & 0 & -1 \\ 0 & 1 & 0 \\ 1 & 0 & 0 \end{bmatrix} \quad z = \boxed{0}$$

Prin procesarea cu aceşti operatori asupra unei imagini binare cu un obiect (pixeli cu valori +1) se realizează ştergerea, pe direcţia specificată, a unui pixel din obiect. De exemplu, (figura 5.8e) după porţiunea de traiectorie planificată A-D, pentru realizarea porţiunii de traiectorie D-G, imaginea de start are ca pixel activ numai punctul D, care se extrage din segmentul (obiectul) A-D, prin procesarea succesivă cu dele.tem.

5.3.4 Planificarea traiectoriei prin determinarea direcţiei optime la fiecare pas

Potrivit metodei prezentate în paragraful precedent, alegerea unei noi direcţii se realizează numai dacă prin menţinerea direcţiei curente robotul nu s-ar mai apropia de ţintă (figura 5.6). Astfel, se reduce la minim numărul de viraje, dar traiectoria planificată nu este cea cu lungimea minimă. În exemplul prezentat, între cele două viraje, punctele A şi D, există punctul B din care direcţia optimă în loc de C ar fi C`. De aceea, pentru a obţine traiectoria optimă din punct de vedere al lungimii drumului de parcurs pentru robot, trebuie să se aleagă direcţia optimă la fiecare pas corespunzător fiecărui pixel din imaginea cu niveluri de gri, de evaluare a distanţei punctelor mediului de lucru faţă de ţintă. Alegerea direcţiei optime la fiecare pas se realizează pe baza principiului prezentat la paragraful 5.3.2. În acest fel, rezultă traiectoria planificată, optimă din punct de vedere al lungimii pentru aplicaţia prezentată în acest capitol (figura 5.9b,c). Vârful săgeţilor indică faptul că alegerea direcţiei optime se realizează la fiecare pas corespunzător discretizării spaţiale a imaginii mediului cu obstacole.

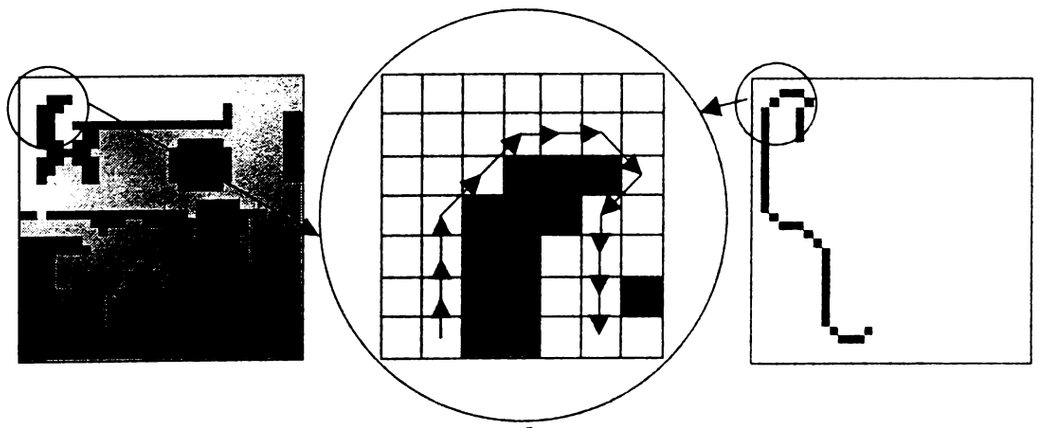


Figura 5.9. Planificarea traiectoriei prin determinarea direcţiei optime la fiecare pas:
(a) imaginea generată pentru estimarea distanţei dintre poziţia start şi poziţia ţintă;
(b) imaginea în detaliu a unei porţiuni din traiectoria planificată optimă;
(c) traiectoria planificată optimă din punct de vedere al lungimii.

5.4 Simularea algoritmul CNN de planificare a traiectoriei unui robot mobil într-un mediu cu obstacole

Pentru testarea algoritmului de planificare a traiectoriei unui robot mobil într-un mediu de lucru cu obstacole utilizând rețele celulare neuronale s-a utilizat mediul de simulare "VisMouse-SimCNN" (Visual mouse platform and multi-layer CNN simulator [124]). Rezultatele experimentale au fost obținute prin simulare cu ajutorul unor programe realizate în limbaj de asamblare "AMC" (Extended analogic macro code and interpreter [130]), (anexa 5, anexa 6).

Timpul total minim de prelucrare necesar algoritmului se obține din timpii minimi necesari pentru prelucrare, la fiecare etapă în parte. Pentru evaluarea distanțelor punctelor spațiului de lucru față de poziția țintă. timpul minim de prelucrare depinde de distanța dintre poziția de start şi poziția de țintă. deoarece este absolut necesar ca unda cu originea în țintă să se propage până în poziția de start.

În etapa de deplasare pe o direcție aleasă, timpul minim necesar este proporțional cu lungimea traseului care trebuie parcurs la planificarea traiectoriei. Din punctul de vedere al timpului total are importanță şi numărul de viraje, adică numărul de puncte în care se face o nouă alegere a direcției optime în vederea atingerii țintei. În cazul determinării direcției optime la fiecare pas, timpul necesar de calcul este mai mare decât la varianta unui număr minim de viraje, dar traiectoria planificată are lungimea minimă. Dintre cele două metode propuse, se poate alege cea convenabilă numai după cunoaşterea şi evaluarea condițiilor impuse şi a scopului urmărit în aplicația concretă.

Pentru ca algoritmul CNN de planificare a traiectoriei unui robot mobil să poată funcționa corect trebuie ca fiecărui punct din spațiul de lucru să i se poată asocia o valoare în etapa de evaluare a distanțelor punctelor față de poziția țintă. În consecință, trebuie avut în vedere domeniul valorilor elementelor de imagine CNN şi faptul că, în prezent, valorile acestea se pot coda pe 8 biți pe un chip-CNN, ceea ce înseamnă că este limitată valoarea minimă a distanței unitare β (figura 5.4). De asemenea, o condiție necesară pentru buna funcționare a algoritmului prezentat este ca la fiecare imagine achiziționată să existe şi să se poată identifica atât punctul de start cât şi punctul țintă. Astfel, rezultă dimensiunile imaginii şi pasul de eşantionare minim pentru discretizarea spațială a imaginii achiziționate.

5.5 Concluzii

Pentru planificarea traiectoriei unui robot mobil într-un mediu cu obstacole, s-a prezentat în acest capitol un algoritm original utilizând rețele neuronale celulare.

Algoritmul CNN de planificare a traiectoriei unui robot într-un mediu cu obstacole, cu ajutorul unei camere video, a fost verificat integral prin simulare.

Rezultatele obținute la testarea algoritmului CNN, propus de autor, confirmă că, prin utilizarea rețelelor neuronale celulare la prelucrarea imaginilor se poate planifica în timp real traiectoria unui robot mobil într-un mediu cu obstacole. De aceea, această metodă de planificare bazată pe imagini constituie o alternativă avantajoasă față de alte procedee utilizate la navigarea robotului mobil. Ocolind obstacolele din mediu, algoritmul CNN propus, furnizează robotului traiectoria planificată optimă din punct de vedere al lungimii şi a numărului de viraje, între poziția de start şi poziția țintă, ori se semnalează dacă ținta nu este accesibilă.

Îmbunătățirea performanțelor algoritmului global CNN de planificare a traiectoriei unui robot, bazată pe imagini, este dependentă de posibilitatea creşterii în continuare a vitezei de prelucrare a imaginilor, care la rândul ei depinde de posibilitatea implementării şi performanțele chipului CNN.

6. Concluzii

6.1 Prezentare sintetică a tezei

În prezenta teză de doctorat sunt studiate posibilitățile de utilizare a rețelelor neuronale celulare la prelucrarea semnalelor bidimensionale, cu aplicații pentru comanda adaptivă a roboților pe baza informației vizuale.

Este evident, ca prin includerea senzorului vizual în structura de comandă a robotului creşte gradul de adaptabilitate al acestuia la modificările din mediul de lucru, fapt ce conduce la extinderea domeniilor de aplicabilitate [50,54,80,110]. Pentru ca această comandă să se poată realiza în timp real, este necesară o putere de calcul deosebit de ridicată, mai ales pentru prelucrarea rapidă şi robustă a imaginii. În lucrare se arată că reducerea timpului de prelucrare şi eliminarea sensibilității sistemului la erorile de calibrare ale camerei video sunt avantajele principale ale comenzii bazate pe imagini față de controlul bazat pe poziții. Dacă reacția vizuală a sistemului este asimptotic stabilă, eroarea de imagine şi în consecință şi eroarea cinematică vor tinde spre zero [4,27,54,58,110].

Ca o altă alternativă față de tehnicile convenționale, rețelele neuronale fiind mijloace adaptive de procesare paralele şi distribuită, pot fi utilizate la rezolvarea problemelor specifice de la comanda adaptivă a robotului [67,73,82], dar şi la controlul pe baza informației vizuale [4,26,107]. S-a arătat că prin utilizarea rețelelor neuronale se poate realiza o poziționare vizuală cu un grad de acuratețe similar cu cel obținut prin tehnici analitice.

Utilizarea rețelelor neuronale celulare în scopul realizării comenzii bazate pe imagini a roboților, se justifică în primul rând, dacă se au în vedere rezultatele care s-au obținut cu aceste rețele la prelucrarea semnalelor bidimensionale, începând de la prelucrări elementare ale imaginilor [1,66,86,71], recunoaştere de forme [93], codarea imaginilor [63,98], realizarea de memorii asociative [3,101], metode de prelucrare a imaginilor bazate pe morfologia matematică [120] şi până la prelucrări complexe spațiio-temporale [28,95].

Rețeaua neuronală celulară de bază (CNN-Cellular neural network [7,8]) este o structură bidimensională rectangulară, fiind formată din circuite analogice neliniare

identice dispuse regulat, numite celule. Având în vedere structura bidimensională specifică a acestei rețele rezultă corespondența imediată a fiecărei celule cu un element discret de imagine. Aceste celule interacționează local între ele pe baza unor operatori (template) A, B şi z care au semnificația unor matrice de pondere. Operația pe care o realizează o rețea neuronală celulară asupra unei imagini de intrare $\boldsymbol{U}(t_0)$ şi imagini inițiale de stare $\boldsymbol{X}(t_0)$, pentru obținerea unei imagini de ieşire stabilă $\boldsymbol{Y}$, este complet definită de aceşti operatori care alcătuiesc o instrucțiune elementară CNN.

Calculatorul universal CNN (CNN-Universal Machine [87]) s-a obținut adăugând la rețeaua neuronală celulară de bază unități de memorie locală, unități de calcul aritmetic şi logic respectiv o unitate de programare globală. Prin implementarea în tehnologia VLSI a acestor calculatoare CNN, rezultă în mod real prelucrarea complet paralelă a semnalelor cu rețelele neuronale celulare. Testarea prelucrărilor pe chipurile CNN din ce în ce mai performante, evidențiază avantajul major al procesării CNN complet paralele, care este realizabilă în timp real [21,57,68,78,79]. De aceea, din punctul de vedere al vitezei de calcul, pentru comanda adaptivă a roboților bazată pe imagini, prelucrarea semnalelor cu rețele neuronale celulare poate constitui o alternativă avantajoasă față de procesarea numerică a semnalelor cu calculatoarele seriale, ori față de alte mijloace hard specializate de procesare paralelă.

În anumite cazuri funcția dorită a se realiza prin prelucrarea semnalelor bidimensionale cu rețele neuronale celulare se obține proiectând în mod corespunzător setul de operatori. Astfel, în capitolul 4 se propun metode noi, originale, de interpolare a semnalelor bidimensionale, folosind rețele neuronale celulare cu operatori liniari de reacție, de dimensiune 3*3, care s-au obținut pe cale analitică.

În general, în cazul unor probleme complexe este dificil de proiectat un singur operator, liniar de dimensiune 3*3, care să ofere soluție potrivită printr-un singur pas de prelucrare. De aceea, problema complexă se va descompune în probleme parțiale care deja se pot soluționa cu câte un singur operator elementar şi în timp foarte scurt, rezultând un algoritm CNN. Pentru a se asigura prelucrarea dorită cu rețeaua neuronală celulară, pe baza unor algoritmi, se realizează programe sau subrutine CNN cu ajutorul a mai multor instrucțiuni elementare (un set de operatori) [20,63,66,92,98,100,109].

În acest fel, rezultă algoritmul CNN original, de urmărire a unui obiect în mişcare, pe baza imaginilor achiziționate cu ajutorul unei camere video montată pe brațul unui robot, prezentat în capitolul 3 şi noul algoritm CNN pentru determinarea punctului

central respectiv a poziției unui obiect dintr-o imagine. Pentru planificarea bazată pe imagini a traiectoriei unui robot mobil într-un mediu cu obstacole, s-a propus în capitolul 5 algoritmul original, utilizând rețele neuronale celulare.

Procedeul CNN se poate considera ca fiind cel mai eficient, pentru o problemă concretă, față de o altă soluție posibilă, numai după implementarea operatorilor şi algoritmilor pe un chip CNN şi funcționarea lor robustă. De aceea, la elaborarea noilor operatori şi algoritmi CNN, prezentați în această lucrare, în vederea validării, s-a ținut cont permanent de posibilitatea implementării acestora pe cel mai performant chip CNN existent [21,68,].

6.2 Contribuții originale

Prelucrarea semnalelor cu rețelele neuronale celulare constituie actualmente un domeniu tehnic în plină dezvoltare. Prin prezentarea generală din capitolul 2 a acestor rețele, s-a urmărit o introducere în domeniu, corespunzător stadiului actual privind noțiunile teoretice şi performanțele obținute în diverse aplicații. Cu acest prilej, pe baza unui material bibliografic bogat şi de strictă actualitate, autorul tezei aduce contribuții legate de sistematizarea principalelor noțiuni în domeniu şi de stabilire unui însemnat număr de concluzii utile cercetării viitoare din teză.

În capitolul 3 este prezentat un algoritm CNN original, de urmărire a unui obiect în mişcare, pe baza imaginilor achiziționate cu ajutorul unei camere video montată pe brațul unui robot [28,30,32]. Acest algoritm a fost verificat integral prin simulare şi testat parțial pe un chip CNN cu 20*22 celule (cP400 [21]). Testarea s-a făcut pentru diferite condiții inițiale care au putut fi stabilite pentru un sistem robotic concret, utilizat în experimentări.

În cadrul acestui capitol, autorul tezei aduce următoarele contribuții originale:

- A elaborat un algoritm CNN de comandă vizuală bazată pe imagini pentru un braț robotic cu două grade de libertate. Algoritmul permite comanda brațului robotic astfel încât acesta să poată urmări un obiect aflat în mişcare, obiect a cărui imagine, utilizată cu rol de comandă, este prelevată cu ajutorul unei camere video montată pe brațul robotului.

  Prin prelucrarea complet paralelă, cu rețele neuronale celulare, a semnalelor bidimensionale, se asigură efectuarea tuturor procesărilor necesare între două

imagini de intrare succesiv achiziționate, fapt ce asigură funcționarea în timp real a sistemului.

- A elaborat operatorul follow.tem, cu ajutorul căruia poate fi extras un obiect având o anumită poziție în imaginea curentă, pe baza desemnării obiectului într-o altă imagine, în care poziția acestuia poate fi diferită față de poziția curentă. Este posibilă în acest fel stabilirea și urmărirea poziției unui obiect aflat în mișcare, într-o succesiune de imagini achiziționate privind obiectul respectiv.
- A fost introdusă noțiunea de imagine de comandă necesară comenzii vizuale bazată pe imagini a cuplelor cinematice ale unui braț robotic.

  Pentru diferite tipuri de imagini de comandă asociate cuplelor cinematice, a stabilit natura și modalitatea de elaborare cu metode CNN a acestora. Proprietățile imaginilor de comandă depind de specificul aplicației concrete și de particularitățile circuitului CNN pe care se implementează algoritmul.

Pe baza analizei comportamentului sistemului robotic studiat și în urma rezultatelor obținute prin simulare respectiv implementare pe un chip CNN a algoritmului de urmărire a unui obiect în mișcare, se poate considera că rețelele neuronale celulare constituie mijloace eficiente, rapide și utile în implementarea comenzii vizuale a unui braț robotic.

- Utilizând rețele neuronale celulare, a elaborat un algoritm CNN pentru determinarea punctului central respectiv a poziției unui obiect dintr-o imagine.
- Pentru determinarea unor proprietăți globale, cum ar fi axele de simetrie sau punctul central al unui obiect, a introdus noțiunea de imagine de măsurare.

Algoritmul pentru determinarea punctului central al unui obiect utilizând rețele neuronale celulare [33,36], folosește numai procesări elementare logice și cu operatori liniari de dimensiune 3*3, fapt ce permite implementarea acestuia cu ușurință pe un chip CNN cu 64*64 de celule (cP4000 [68]). În acest fel prelucrarea se poate efectua în timp real, datorită procesării complet paralele și ca urmare a reducerii substanțiale a numărului de încărcări pe chip a imaginilor inițiale. Acest algoritm, deși furnizează soluția după mai mulți pași de procesări elementare, prezintă avantajul că timpul total de procesare nu depinde de dimensiunea obiectului la care se caută punctul central. Totodată, în cazul acestui algoritm, rezultatul nu este afectat de erorile introduse la determinarea punctului central cu operatorul center.tem [125], în cazul în care obiectul are orificii sau nu este cu contur închis.

În capitolul 4, în totalitate original, sunt prezentate modalități noi de interpolare cu rețele neuronale celulare a semnalelor bidimensionale [31,37,39,44].

Contribuțiile originale prezente în acest capitol sunt următoarele:

- A determinat pe cale analitică operatorii de reacție de tip A, liniari si de dimensiune 3*3, aintpol1.tem, aintpol2.tem si aintpol3.tem, care permit interpolarea CNN a semnalelor bidimensionale.

  Aceşti operatori, cu aplicabilitate atât în cazul imaginilor binare cât şi a imaginilor cu niveluri de gri, prezintă avantajul că pot fi implementați pe chipuri CNN. Față de procedeele de interpolare existente, chiar şi cele bazate pe rețele neuronale celulare, interpolarea cu operatorii nou propuşi se poate efectua în timp real, ca urmare a reducerii substanțială a numărului necesar de încărcări pe chip a imaginilor inițiale şi deci a duratei totale de calcul. Datorită procesării complet paralele, timpul de execuție nu creşte practic odată cu creşterea dimensiunilor imaginii.
- A definit setul necesar şi suficient de condiții inițiale şi de frontieră ale rețelei neuronale celulare, pentru determinarea pe cale analitică operatorilor de reacție de tip A, în vederea prelucrării de imagini cu niveluri de gri.
- Pentru dimensionarea operatorilor de interpolare a utilizat un procedeu original, care are la bază minimizarea unei funcții de cost concomitent cu condiția asigurării convergenței stării celulelor către o stare stabilă.

  Metoda propusă oferă şi o soluție la problema proiectării operatorilor de reacție de tip A, pentru prelucrarea unor imagini cu niveluri de gri.

Funcționarea corectă a acestor noi metode de interpolare CNN s-a verificat prin simulare şi testare pe un chip CNN de 64*64 de celule (cP4000 [68]). Interpolarea semnalelor bidimensionale cu rețele neuronale, utilizând algoritmii propuşi în acest capitol, prezintă interes aplicativ nu numai în robotică dar şi la mărirea şi interpretarea zonelor de interes major din imaginile biomedicale respectiv la obținerea de imagini de calitate HDTV în cazul recepției semnalului TV analogic uzual, cu păstrarea dimensiunilor elementului de imagine original.

Având în vedere viteza de procesare ridicată, metoda de interpolare propusă poate servi la obținerea imaginilor de comandă ale unui robot industrial, când se cunosc deja valorile ale unor elemente de imagine corespunzătoare pentru anumite puncte de referință din spațiu.

Interpolarea CNN în timp real cu operatorii propuşi poate constitui o modalitate eficientă şi rapidă la reconstrucţia unor imagini deteriorate ori cunoscute parţial sau la codarea şi decodarea imaginilor în vederea transmiterii acestora.

Capitolul 5, în totalitate original, tratează problema planificării traiectoriei prescrise a unui robot mobil într-un mediu cu obstacole, utilizând reţele neuronale celulare [35,38]. Algoritmul realizează planificarea traiectoriei prin procesarea imaginii mediului în care navighează autonom robotul mobil. Algoritmul nou propus furnizează robotului o traiectorie prescrisă între poziţia de start şi o poziţie ţintă, optimă din punctul de vedere al lungimii traseului şi a numărului de viraje, ori se semnalează faptul că ţinta nu este accesibilă.

Acest capitol conţine următoarele contribuţiile originale:

- Pentru un robot mobil care se deplasează autonom într-un mediu cu obstacole, a fost elaborat un algoritm analogic global, bazat pe reţele neuronale celulare, de planificare a traiectoriei prescrise cu evitarea obstacolelor.
- Metoda de planificare a traiectoriei prescrise a fost optimizată printr-un procedeu CNN, încât traiectoria să prezinte un număr minim de viraje.
- A elaborat familia de operatori del.tem care corespund direcţiilor N, S, E, V, S-E, N-E, N-V, S-V. Procesarea cu aceşti operatori asupra unei imagini binare cu un obiect (pixeli cu valori +1) are ca rezultat ştergerea, pe direcţia specificată, a unui pixel din obiect. Prin utilizarea succesivă a acestor operatori pot fi extrase elementele de imagini de la extremităţile unui obiect, după o direcţie.
- A fost elaborat un procedeu CNN prin care se poate determina traiectoria optimă, astfel încât aceasta să fie de lungime minimă.

Rezultatele obţinute la testarea algoritmului CNN propus justifică afirmaţia că prin utilizarea reţelelor neuronale celulare se poate determina în timp real traiectoria prescrisă unui robot mobil atunci când acesta navighează autonom într-un mediu cu obstacole. Această metodă de planificare se bazează exclusiv pe prelucrare de imagini şi constituie o alternativă avantajoasă faţă de alte procedeele utilizate la navigarea roboţilor mobili.

6.3 Generalizări şi direcţii de cercetare viitoare

Conducerea adaptivă a roboţilor pe baza informaţiei vizuale şi utilizarea reţelelor neuronale la prelucrarea semnalelor bidimensionale continuă să fie în centrul atenţiei cercetătorilor din aceste două domenii [70,87,115].

Pe baza rezultatelor obţinute în prezenta lucrare, se poate afirma că reţelele neuronale constituie o alternativă eficientă de prelucrarea în timp real a semnalelor pentru comanda adaptivă a robotului pe baza informaţiei vizuale. Soluţiile oferite pentru algoritmului CNN de urmărire a unui obiect în mişcare, permit implementarea imediată pe un chip CNN de 64*64 de celule (cP4000 [68]), ori pe alte circuite CNN mai performante care vor apare în viitor. Astfel se vor îmbunătăţi şi performanţele algoritmului, deoarece timpul de prelucrare nu creşte proporţional cu mărirea dimensiunilor imaginii prelucrate.

Extinderea comenzii se poate realiza şi pentru un braţ robotic cu trei grade de libertate. Pentru acest caz cât şi pentru conducerea uni robot mobil se pot utiliza rezultatele obţinute cu reţelele neuronale celulare la detecţia de adâncime prin interpretarea imaginilor stereo [84,116]. Soluţia potrivită pentru braţul robotic cu trei grade de libertate poate fi conducerea vizuală 2-1/2D [70], prin utilizarea a două camere video în domeniul CNN.

Creşterea performanţelor algoritmului CNN utilizat la comanda vizuală a robotului este posibilă şi prin utilizarea în algoritm a memoriilor asociative CNN [43,101] respectiv prin fuziunea semnalelor care provin şi de la alţi senzori (tactili sau cei de forţă/moment) [75].

În toate metodele de conducere prezentate, prelucrarea rapidă şi robustă a imaginii prezintă o deosebită importanţă. Ca urmare, se impune în cercetările viitoare, ca algoritmii bazaţi pe procesări de suprafeţe şi care sunt sensibili la schimbări de fundal, să fie înlocuiţi cu metode bazate pe procesări de muchii, mai rapide şi mai robuste [66,69]. Metoda utilizată pentru prelucrarea de imagini trebuie să aibă în vedere însă şi natura aplicaţiei concrete [26].

La elaborarea algoritmului CNN de comandă a braţului robotic, propus în această lucrare, s-a luat în considerare cazul cel mai simplu, al comenzii cinematice, studiul comportamentului dinamic fiind încă o problema deschisă care ar putea fi abordată în

viitor. În această privință se impune și abordarea cu metode CNN a comenzii vizuale directe bazată pe imagini a robotului [15,50,54], realizabilă mai ales dacă circuitele CNN permit utilizarea de operatori variabili în spațiu și în timp [57].

Respectând condițiile inițiale specificate, metoda propusă de interpolarea CNN poate fi utilă la determinarea pe cale analitică a operatorilor CNN și în alte aplicații în care se utilizează imagini cu niveluri de gri, dacă se poate identifica o funcție de cost potrivită. Astfel, pe baza metodei prezentate se poate extinde interpolarea CNN și pentru spațiul 3D. Asemenea prelucrări se pot dovedi utile la reconstrucția suprafețelor pe baza imaginilor stereo sau la interpolarea între straturi. În aceste cazuri a treia variabilă a funcției imagine este coordonata din spațiu z iar dacă interpolarea se realizează între cadrele imaginii, cea de a treia coordonată a imaginii este variabila timp.

Îmbunătățirea performanțelor algoritmului global CNN de conducere a unui robot mobil într-un mediu cu obstacole este dependentă de posibilitatea creșterii în continuare a vitezei de prelucrare a imaginilor. Creșterea vitezei de procesare depinde la rândul ei de posibilitatea implementării algoritmului pe un chip CNN și performanțele chipului.

Precizia de poziționare a robotului în mediul de lucru pe baza informației vizuale, se poate crește dacă o dată cu apropierea robotului de țintă se va putea modifica dimensiunea imaginilor achiziționate. Acest deziderat poate fi realizat prin reglarea adecvată a poziției și a distanței focale a camerei video, încât să fie achiziționată numai imaginea corespunzătoare zonei de interes din mediul de lucru. În acest sens, o posibilă soluție o poate reprezenta îmbinarea algoritmului CNN de urmărire a unui obiect cu o cameră video, prezentat în capitolul trei, cu algoritmul de planificarea traiectoriei, propus în capitolul cinci.

Cercetările privind planificarea traiectoriei unui robot mobil care navighează autonom într-un mediu cu obstacole, pot fi extinse în cel puțin următoarele două direcții.

Într-un mediu cu obstacole aflate în mișcare rețelele neuronale se pot utiliza și pentru predicția evoluției mediului [6,64]. Adaptabilitatea robotului la navigarea în mediu crește dacă la planificarea traiectoriei se ține cont și de traiectoria estimată pentru obstacolele în mișcare.

Pentru creșterea adaptabilității la mediu a robotului mobil, la navigarea prin metodă locală este luată uneori în considerare și dinamica robotului [25,115]. Rețelele neuronale celulare își dovedesc în acest caz utilitatea și în ce privește rezolvarea ecuațiilor diferențiale care intervin în studiu.

Tehnicile analogice implementate cu rețele neuronale celulare constituie o nouă viziune asupra algoritmilor şi metodelor de prelucrare din domeniului procesării spațiio-temporale a semnalelor. Utilizarea lor practică în viitor se va face cu precădere pentru soluționarea în timp real a unor probleme complexe.

Anexa 1

```
; Program AMC pentru testarea algoritmului CNN de urmarire
;cu o camera video, a unui obiect aflat in miscare

start:
      ;host.cam.start 1
      ;wait 100
      ;host.start.frgb DT3153
      ;host.mov.frgb.pic LAM1 -0.5 92 68 64 64
      host.load.tem follow.tem TEM1
      host.load.tem hollow.tem TEM2
      host.load.tem center1.tem TEM3
      host.load.tem center2.tem TEM4
      host.load.tem center3.tem TEM5
      host.load.tem center4.tem TEM6
      host.load.tem center5.tem TEM7
      host.load.tem center6.tem TEM8
      host.load.tem center7.tem TEM9
      host.load.tem center8.tem TEM10
      host.load.tem treshold.tem TEM11
      ;numar imagini de intrare din memorie
      mov.gam.gam 1 GAM6
      mov.gam.gam 7 GAM7
      mov.gam.gam 1 GAM3
      ;imagini de memorare initiale
      host.load.pic mask7.bmp LAM2
      ;imagini de comanda
      ;imagini binare pentru teta1 si teta 2
      host.load.pic comtetab1.bmp LAM7
      host.load.pic comtetab2.bmp LAM8
      host.display LAM7 7
      host.display LAM8 8
foll:
```

```
      ;numar cicluri pentru center.tem
       mov.gam.gam 0 GAM4
      mov.gam.gam 18 GAM5
      ;imagine de memorare curenta
      host.display LAM2 2
      ;imagine de intrare curenta cu niveluri de gri
      host.loadn.pic moobj70000.bmp GAM6 LAM6
      mov.lam.lam LAM6 LAM1
      host.display LAM1 1
      ar.tem TEM11 LAM6 LAM6 LLM1 5 0 0 NIL NIL;treshold
      host.display LLM1 13
      ;achizitionarea cu camera video
      ;de imagini cu niveluri de gri
      ;host.mov.frgb.pic LAM6 gray 92 68 64 64
      ;host.display LAM6 6
      ;achizitionarea cu camera video de imagini binare
      ;host.mov.frgb.pic LAM1 -0.5 92 68 64 64
      ar.tem TEM1 LLM1 LAM2 LAM3 50 -1 -1 NIL NIL;follow
      host.display LAM3 3
      ar.tem TEM2 LAM3 LAM3 LAM4 50 -1 -1 NIL NIL;hollow
      mov.lam.lam LAM3 LAM2
      mov.lam.lam LAM4 LAM2
      host.display LAM4 4
      mov.lam.lam LAM4 LAM5
loop:
      call center
      sc.inc.gam GAM4                ;bucla pentru center
      sc.rel.gt.gam GAM5 GAM4 GLM4
      host.display LAM5 5
      jumpc GLM4 loop
      host.display LAM5 5
      mov.lam.lam LAM4 LAM9
      mov.lam.lam LAM4 LAM10
      ;host.display LAM9 4
```

```
      ;host.display LAM10 4
      ar.mul.lam LAM9 LAM7 LAM9     ;calcul teta1
      host.display LAM9 9
      conv.lam.gam.avg LAM9 GAM1
      conv.gam.lam GAM1 LAM9
      host.display LAM9 10
      ar.mul.lam LAM10 LAM8 LAM10   ;calcul teta2
      host.display LAM10 11
      conv.lam.gam.avg LAM10 GAM2
      conv.gam.lam GAM2 LAM10
      host.display LAM10 12
      ;determinarea directiei de miscare
      ;mov.gam.gam 0 GAM1           ;right-left
      ;mov.gam.gam 0 GAM2           ;up-down
      sc.rel.gt.gam -0.03 GAM1 GLM2
      sc.rel.lt.gam 0.03  GAM1 GLM1
      sc.rel.gt.gam -0.03 GAM2 GLM4
      sc.rel.lt.gam 0.03  GAM2 GLM3
      ;subrutina de comanda a camerei video
      call comcam
      sc.inc.gam GAM6               ; conditia de stop
      sc.rel.gt.gam  GAM7 GAM6 GLM6
      jumpc GLM6 foll
end:
      ;host.stop.frgb
      ;host.cam.stop
      end
;Subrutina center
center:
      ar.tem TEM3 LAM5 LAM5 LAM5 GAM3 -1 -1 NIL NIL;center1
      ar.tem TEM4 LAM5 LAM5 LAM5 GAM3 -1 -1 NIL NIL;center2
      ar.tem TEM5 LAM5 LAM5 LAM5 GAM3 -1 -1 NIL NIL;center3
      ar.tem TEM6 LAM5 LAM5 LAM5 GAM3 -1 -1 NIL NIL;center4
      ar.tem TEM7 LAM5 LAM5 LAM5 GAM3 -1 -1 NIL NIL;center5
```

```
      ar.tem TEM8 LAM5 LAM5 LAM5 GAM3 -1 -1 NIL NIL;center6
      ar.tem TEM9 LAM5 LAM5 LAM5 GAM3 -1 -1 NIL NIL;center7
      ar.tem TEM10 LAM5 LAM5 LAM5 GAM3 -1 -1 NIL NIL;center8
       ret
;Subrutina de comanda a camerei video
comacam:
      jumpc GLM1 conright
      jumpc GLM2 conleft
      jumpc GLM3 up
      jumpc GLM4 down
      jump contin
conright:                          ;GLM1
      jumpc GLM3 rightup           ;GLM1,GLM3
      jumpc GLM4 rightdown         ;GLM1,GLM4
      jump right                   ;GLM2
conleft:                           ;GLM2
      jumpc GLM3 leftup            ;GLM2,GLM3
      jumpc GLM4 leftdown          ;GLM2,GLM4
      jump left                    ;GLM2
zero:
      host.cam.mov.rel 0x0000 0x0000 0x1c;zeromove
      jump contin
right:
      host.cam.mov.rel 0x0002 0x0000 0x1c;right
      jump contin
up:
      host.cam.mov.rel 0x0000 0x0002 0x1c;up
      jump contin
rightup:
      host.cam.mov.rel 0x0002 0x0002 0x1c;rightup
      jump contin
left:
      host.cam.mov.rel 0xfffd 0x0000 0x1c;left
      jump contin
```

```
down:
      host.cam.mov.rel 0x0000 0xfffd 0x1c;down
      jump contin
leftup:
      host.cam.mov.rel 0xfffd 0x0002 0x1c;leftup
      jump contin
rightdown:
      host.cam.mov.rel 0x0002 0xfffd 0x1c;rightdown
      jump contin
leftdown:
      host.cam.mov.rel 0xfffd 0xfffd 0x1c;leftdown
      jump contin
contin:
      wait 200
      ret
```

Anexa 2

```
; Program AMC pentru testarea algoritmului CNN
;de urmarire in timp real cu o camera video
;a unui fascicul laser in miscare

start:
      host.cam.start 2
      host.start.frgb DT3153
      wait 500
      host.mov.frgb.pic LAM9 -0.5 150 60 300 300
      wait 500
      ;imagini binare pentru teta1 si teta 2
      host.load.pic comt31.bmp LLM7
      host.load.pic comt32.bmp LLM8
      ar.not.llm LLM7 LLM7
      ar.not.llm LLM8 LLM8
      ;numar de cadre de imagine achizitionate
       mov.gam.gam 1 GAM5
       mov.gam.gam 127 GAM6
movobj:
      ;achizitionarea cu camera video
      ;de imagini cu niveluri de gri
      host.mov.frgb.pic LAM10 gray 150 60 300 300
      host.display LAM10 10
      ;achizitionarea cu camera video de imagini binare
      host.mov.frgb.pic LLM11 -0.95 150 60 300 300
      ;host.display LLM11 11
      ;memorarea imaginilor achizitionate cu camera video
      ;host.saven.pic  moobj30000.bmp GAM5 LAM10
      ar.xor.llm LLM11 LLM7 LLM15   ; calcul teta1
      ;host.display LLM15 8
      conv.llm.gam.cnt LLM15 GAM1
```

```
        ar.xor.llm LLM11 LLM8 LLM16   ; calcul teta2
        ;host.display LLM16 9
        conv.llm.gam.cnt LLM16 GAM2
        ;determinarea directiei de miscare
        sc.rel.gt.gam  44990 GAM1  GLM2;45000
        sc.rel.lt.gam  45010 GAM1  GLM1
        sc.rel.gt.gam  44990 GAM2  GLM4
        sc.rel.lt.gam  45010 GAM2  GLM3
        ;subrutina de comanda a camerei video
        call comcam
        sc.inc.gam GAM5
        sc.rel.gt.gam  GAM6 GAM5 GLM5
        jumpc GLM5 movobj
end:    host.stop.frgb
        host.cam.stop
        end
;Subrutina de comanda a camerei video
comacam:
        jumpc GLM1 conright
        jumpc GLM2 conleft
        jumpc GLM3 up
        jumpc GLM4 down
        jump contin
conright:                           ;GLM1
        jumpc GLM3 rightup          ;GLM1,GLM3
        jumpc GLM4 rightdown        ;GLM1,GLM4
        jump right                  ;GLM2
conleft:                            ;GLM2
        jumpc GLM3 leftup           ;GLM2,GLM3
        jumpc GLM4 leftdown         ;GLM2,GLM4
        jump left                   ;GLM2
zero:
        host.cam.mov.rel 0x0000 0x0000 0x1c;zeromove
        jump contin
```

```
right:
      host.cam.mov.rel 0x0002 0x0000 0x1c;right
      jump contin
up:
      host.cam.mov.rel 0x0000 0x0002 0x1c;up
      jump contin
rightup:
      host.cam.mov.rel 0x0002 0x0002 0x1c;rightup
      jump contin
left:
      host.cam.mov.rel 0xfffd 0x0000 0x1c;left
      jump contin
down:
      host.cam.mov.rel 0x0000 0xfffd 0x1c;down
      jump contin
leftup:
      host.cam.mov.rel 0xfffd 0x0002 0x1c;leftup
      jump contin
rightdown:
      host.cam.mov.rel 0x0002 0xfffd 0x1c;rightdown
      jump contin
leftdown:
      host.cam.mov.rel 0xfffd 0xfffd 0x1c;leftdown
      jump contin
contin:
      wait 200
      ret
```

Anexa 3

```
; Program AMC pentru testarea algoritmului CNN
;de determinare a punctului central al unui obiect

start:
      mov.gam.gam 1 GAM1
      host.load.tem filwhite.tem TEM1
      host.load.tem filblack.tem TEM2
      host.load.tem shadsimud.tem TEM3
      host.load.tem shadsimrl.tem TEM4
      ;imagine binara cu un obiect
      host.load.pic obiect3.bmp LLM4
      ;imaginea de masurare x
      host.load.pic imcantx.bmp LAM1
      ;imaginea de masurare y
      host.load.pic imcanty.bmp LAM9
      ar.nand.llm LLM4 LLM4 LLM5
      host.display LLM5 1
      host.display LAM1 2
      host.display LAM9 3
      ;determinarea axei de simetrie dupa directia x
ar.tem TEM1 LAM1 LAM1 LAM2 GAM1 zeroflux zeroflux LLM4 NIL
      host.display LAM2 4          ;imaginea cu maxim
ar.tem TEM2 LAM1 LAM1 LAM3 GAM1 zeroflux zeroflux LLM4 NIL
      host.display LAM3 5          ;imaginea cu minim
;determinarea valorii maxime si valorii minime dupa axa x
      conv.lam.gam.max LAM2 GAM1
      conv.lam.gam.max LAM2 GAM4
      sc.mul.gam 0.5 GAM4 GAM4
      conv.lam.gam.min LAM3 GAM5
      sc.mul.gam 0.5 GAM5 GAM5
      sc.add.gam GAM4 GAM5 GAM6
```

```
      sc.add.gam -0.01 GAM6 GAM7
      conv.gam.lam GAM7 LAM5
      ar.sub.lam LAM2 LAM5 LAM6
      conv.lam.llm LAM6 LLM7
      host.display LLM7 6
      sc.add.gam 0.01 GAM6 GAM9
      conv.gam.lam GAM9 LAM7
      ar.sub.lam LAM3 LAM7 LAM8
      conv.lam.llm LAM8 LLM8
      ar.nand.llm LLM8 LLM8 LLM20
      ;axa de simetrie dupa directia x
      host.display LLM20 7
      ar.landnot2.llm LLM7 LLM8 LLM9
      host.display LLM9 8
ar.tem TEM3 LLM9 LLM9 LLM9 64 zeroflux  zeroflux NIL NIL
      ;axa prelungita de simetrie dupa directia x
      host.display LLM9 9
      ;determinarea axei de simetrie dupa directia y
ar.tem TEM1 LAM9 LAM9 LAM2 GAM1 zeroflux zeroflux LLM4 NIL
      host.display LAM2 10          ; imaginea cu maxim
ar.tem TEM2 LAM9 LAM9 LAM3 GAM1 zeroflux zeroflux LLM4 NIL
      host.display LAM3 11          ; imaginea cu minim
;determinarea valorii maxime si valorii minime dupa axa y
      conv.lam.gam.max LAM2 GAM1
      conv.lam.gam.max LAM2 GAM4
      sc.mul.gam 0.5 GAM4 GAM4
      conv.lam.gam.min LAM3 GAM5
      sc.mul.gam 0.5 GAM5 GAM5
      sc.add.gam  GAM4 GAM5 GAM6
      sc.add.gam -0.01 GAM6 GAM7
      conv.gam.lam GAM7 LAM5
      ar.sub.lam LAM2 LAM5 LAM6
      conv.lam.llm LAM6 LLM7
      host.display LLM7 12
```

```
        ar.nand.llm LLM7 LLM7 LLM19
        sc.add.gam 0.01 GAM6 GAM9
        conv.gam.lam GAM9 LAM7
        ar.sub.lam LAM3 LAM7 LAM8
        conv.lam.llm LAM8 LLM8
        ar.nand.llm LLM8 LLM8 LLM20
        host.display LLM20 13
        ar.1andnot2.llm LLM7 LLM8 LLM10
        ;axa de simetrie dupa directia y
        host.display LLM10 14
ar.tem TEM4 LLM10 LLM10 LLM10 64 zeroflux zeroflux NIL NIL
        ;axa prelungita de simetrie dupa directia y
         host.display LLM10 15
        ;determinarea punctului central
        ar.and.llm LLM9 LLM10 LLM11
        ;imaginea continand numai punctul central
        ;al obiectului
        host.display LLM11 16
end:  end
```

Anexa 4

```
; Program AMC pentru testarea metodei CNN de interpolare
;cu aintpol1.tem, aintpol2.tem, aintpol3.tem
;se determina eroarea fata de imaginea originala

start:
      ;bucla pentru determinarea timpului de procesare
      ;timpul de procesare CNN
      ;timpul mediu de procesare completa
      mov.gam.gam 1 GAM1
      mov.gam.gam 100 GAM2
      mov.gam.gam 1 GAM3
      host.load.tem aintpol1.tem TEM1
      host.load.tem aintpol2.tem TEM2
      host.load.tem aintpol3.tem TEM3
      ;imaginea care se interpoleaza
      host.load.pic interpol.bmp LAM1
      ;host.load.pic aintpolgsch.bmp LAM1
      ;imaginea originala sau ideala
      host.load.pic ideal.bmp LAM5
      ;host.load.pic aintlena.bmp LAM5
      ;imaginea masca
      host.load.pic mask.bmp LLM4
      ;host.load.pic aintpolmaskgs.bmp LLM4
      host.display LAM1 1
      host.display LAM5 3
      host.display LLM4 2
cyc:
ar.tem TEM1 LAM1 LAM1 LAM2 GAM1 zeroflux zeroflux LLM4 NIL
      host.display LAM2 4
ar.tem TEM2 LAM1 LAM1 LAM3 GAM1 zeroflux zeroflux LLM4 NIL
      host.display LAM3 5
```

```
ar.tem TEM3 LAM1 LAM1 LAM4 GAM1 zeroflux zeroflux LLM4 NIL
      host.display LAM4 6
      ;calculul imaginilor de eroare
      ar.sub.lam LAM5 LAM2 LAM8
      host.display LAM8 7
      ar.sub.lam LAM5 LAM3 LAM9
      host.display LAM9 8
      ar.sub.lam LAM5 LAM4 LAM10
      host.display LAM10 9
      sc.add.gam GAM1 GAM3 GAM1
      sc.rel.gt.gam GAM2 GAM1 GLM1
      jumpc GLM1 cyc
end:  end
```

Anexa 5

```
; Program AMC pentru testarea algoritmului CNN
;de planificare a traiectoriei unui robot mobil,
;la deplasarea între doua puncte ale unui mediu cu
;obstacole, pe baza principiului efectuarii miscarii pe o
;directie aleasa pana se realizeaza apropierea de tinta.

start:
      host.load.tem explore.tem TEM1
      host.load.tem shifte.tem TEM2 ;shift est
      host.load.tem shiftv.tem TEM3 ;shift vest
      host.load.tem shiftn.tem TEM4 ;shift nord
      host.load.tem shifts.tem TEM5 ;shift sud
      host.load.tem shiften.tem TEM6;shift est-nord
      host.load.tem shiftvn.tem TEM7;shift vest-nord
      host.load.tem shiftes.tem TEM8;shift est-sud
      host.load.tem shiftvs.tem TEM9;shift vest-sud
      host.load.tem filwhite.tem TEM10
      host.load.tem selecten.tem TEM11;select est-nord
      host.load.tem selectes.tem TEM12; select est-sud
      host.load.tem selectvn.tem TEM13; select vest-nord
      host.load.tem selectvs.tem TEM14; select vest-sud
      host.load.tem delen.tem TEM15 ;del nord
      host.load.tem deles.tem TEM16 ;del sud
      host.load.tem delvn.tem TEM17 ;del vest-nord
      host.load.tem delvs.tem TEM18 ;del vest-sud
      host.load.tem selecte.tem TEM22;select est
      host.load.tem selects.tem TEM23;select sud
      host.load.tem selectv.tem TEM24;select vest
      host.load.tem selectn.tem TEM25;select nord
      host.load.tem dele.tem TEM26  ;del est
      host.load.tem dels.tem TEM27  ;del sud
```

```
        host.load.tem delv.tem TEM28  ;del vest
        host.load.tem deln.tem TEM29  ;del nord
        ;imaginea mediului
         host.load.pic mediu.bmp LLM1 BIN
        ;imaginea  inversa tinta
        host.load.pic tinta.bmp LLM2 BIN
        ;imaginea  start
        host.load.pic start.bmp LLM3 BIN
        ar.nand.llm LLM2 LLM2 LLM4
        ar.nand.llm LLM3 LLM3 LLM3
        ;imaginea traiectoriei planificate
        mov.llm.llm LLM3 LLM6
        host.display LLM1 1
        ar.nand.llm LLM1 LLM1 LLM1
        host.display LLM3 2
        host.display LLM4 3
;evaluarea distantei pozitiei actuale fata de pozitia tinta
        ar.tem TEM1 LLM1 LLM2 LAM3 200 1 1 LLM1 NIL
        host.display LAM3 4
loop:
        ar.nand.llm LLM3 LLM3 LLM9
;Subprogram pentru evaluarea directiei optime
        call optdir
;deplasarea pe directia optima aleasa
;pana se realizeaza apropierea de tinta
estnord: ar.tem TEM11 LAM3 LLM3 LLM7 30 1 -1 NIL NIL
        ar.or.llm LLM6 LLM7 LLM6
        host.display LLM6 5
        mov.gam.gam 0 GAM9
        mov.gam.gam 30 GAM10
del_en: ar.tem TEM15 LLM3 LLM3 LLM3 1 -1 -1 LLM7 NIL
        sc.inc.gam GAM9
        sc.rel.gt.gam GAM10 GAM9 GLM2
        jumpc GLM2 del_en
```

```
        jump final
estsud: ar.tem TEM12 LAM3 LLM3 LLM7 30 1 -1 NIL NIL
        ar.or.llm LLM6 LLM7 LLM6
        host.display LLM6 5
        mov.gam.gam 0 GAM9
        mov.gam.gam 30 GAM10
del_es: ar.tem TEM16 LLM3 LLM3 LLM3 1 -1 -1 LLM7 NIL
        sc.inc.gam GAM9
        sc.rel.gt.gam GAM10 GAM9 GLM2
        jumpc GLM2 del_es
        jump final
vestnord: ar.tem TEM13 LAM3 LLM3 LLM7 30 1 -1 NIL NIL
        ar.or.llm LLM6 LLM7 LLM6
        host.display LLM6 5
        mov.gam.gam 0 GAM9
        mov.gam.gam 30 GAM10
del_vn: ar.tem TEM17 LLM3 LLM3 LLM3 1 -1 -1 LLM7 NIL
        sc.inc.gam GAM9
        sc.rel.gt.gam GAM10 GAM9 GLM2
        jumpc GLM2 del_vn
        jump final
vestsud: ar.tem TEM14 LAM3 LLM3 LLM7 30 1 -1 NIL NIL
        ar.or.llm LLM6 LLM7 LLM6
        host.display LLM6 5
        mov.gam.gam 0 GAM9
        mov.gam.gam 30 GAM10
del_vs: ar.tem TEM18 LLM3 LLM3 LLM3 1 -1 -1 LLM7 NIL
        sc.inc.gam GAM9
        sc.rel.gt.gam GAM10 GAM9 GLM2
        jumpc GLM2 del_vs
        jump final
est:    ar.tem TEM22 LAM3 LLM3 LLM7 30 1 -1 NIL NIL
        ar.or.llm LLM6 LLM7 LLM6
        host.display LLM6 5
```

```
        mov.gam.gam 0 GAM9
        mov.gam.gam 30 GAM10
del_e:  ar.tem TEM26 LLM3 LLM3 LLM3 1 -1 -1 LLM7 NIL
        sc.inc.gam GAM9
        sc.rel.gt.gam GAM10 GAM9 GLM2
        jumpc GLM2 del_e
        jump final
sud:    ar.tem TEM23 LAM3 LLM3 LLM7 30 1 -1  NIL NIL
        ar.or.llm LLM6 LLM7 LLM6
        host.display LLM6 5
        mov.gam.gam 0 GAM9
        mov.gam.gam 30 GAM10
del_s:  ar.tem TEM27 LLM3 LLM3 LLM3 1 -1 -1 LLM7 NIL
        sc.inc.gam GAM9
        sc.rel.gt.gam GAM10 GAM9 GLM2
        jumpc GLM2 del_s
        jump final
vest:   ar.tem TEM24 LAM3 LLM3 LLM7 30 1 -1  NIL NIL
        ar.or.llm LLM6 LLM7 LLM6
        host.display LLM6 5
        mov.gam.gam 0 GAM9
        mov.gam.gam 30 GAM10
del_v:  ar.tem TEM28 LLM3 LLM3 LLM3 1 -1 -1 LLM7 NIL
        sc.inc.gam GAM9
        sc.rel.gt.gam GAM10 GAM9 GLM2
        jumpc GLM2 del_v
        jump final
nord:   ar.tem TEM25 LAM3 LLM3 LLM7 30 1 -1  NIL NIL
        ar.or.llm LLM6 LLM7 LLM6
        host.display LLM6 5
        mov.gam.gam 0 GAM9
        mov.gam.gam 30 GAM10
del_n:  ar.tem TEM29 LLM3 LLM3 LLM3 1 -1 -1 LLM7 NIL
        sc.inc.gam GAM9
```

```
        sc.rel.gt.gam GAM10 GAM9 GLM2
        jumpc GLM2 del_n
        jump final
final:
        ; atingerea tintei
        ar.nand.llm LLM3 LLM3 LLM5
        ar.xor.llm LLM5 LLM2 LLM8
        host.display LLM8 6
        conv.llm.lam LLM8 LAM1
        conv.lam.gam.avg LAM1 GAM1
        sc.rel.equ.gam -1 GAM1  GLM1
        host.display LLM6 5
        jumpnc GLM1 loop
        end: end
;Subprogram pentru evaluarea directiei optime
optdir:
        ar.tem TEM2 LLM9 LLM9 LLM4 1 1 1 NIL NIL
        ar.tem TEM10 LLM4 LAM3 LAM2 1 1 1 LLM4 NIL
        conv.lam.gam.max LAM2 GAM1
        ar.tem TEM3 LLM9 LLM9 LLM4 1 1 1 NIL NIL
        ar.tem TEM10 LLM4 LAM3 LAM2 1 1 1 LLM4 NIL
        conv.lam.gam.max LAM2 GAM2
        ar.tem TEM4 LLM9 LLM9 LLM4 1 1 1 NIL NIL
        ar.tem TEM10 LLM4 LAM3 LAM2 1 1 1 LLM4 NIL
        conv.lam.gam.max LAM2 GAM3
        ar.tem TEM5 LLM9 LLM9 LLM4 1 1 1 NIL NIL
        ar.tem TEM10 LLM4 LAM3 LAM2 1 1 1 LLM4 NIL
        conv.lam.gam.max LAM2 GAM4
        ar.tem TEM6 LLM9 LLM9 LLM4 1 1 1 NIL NIL
        ar.tem TEM10 LLM4 LAM3 LAM2 1 -1 1 LLM4 NIL
        conv.lam.gam.max LAM2 GAM5
        ar.tem TEM7 LLM9 LLM9 LLM4 1 1 1 NIL NIL
        ar.tem TEM10 LLM4 LAM3 LAM2 1 1 1 LLM4 NIL
        conv.lam.gam.max LAM2 GAM6
```

```
        ar.tem TEM8 LLM9 LLM9 LLM4 1 1 1 NIL NIL
        ar.tem TEM10 LLM4 LAM3 LAM2 1 1 1 LLM4 NIL
        conv.lam.gam.max LAM2 GAM7
        ar.tem TEM9 LLM9 LLM9 LLM4 1 1 1 NIL NIL
        ar.tem TEM10 LLM4 LAM3 LAM2 1 1 1 LLM4 NIL
        conv.lam.gam.max LAM2 GAM8
        ;se determina directia optima
unu:    sc.rel.lt.gam GAM1 GAM2 GLM1
        jumpnc GLM1 doi
        sc.rel.lt.gam GAM1 GAM2 GLM1
        jumpnc GLM1 doi
        sc.rel.lt.gam GAM1 GAM3 GLM1
        jumpnc GLM1 trei
        sc.rel.lt.gam GAM1 GAM4 GLM1
        jumpnc GLM1 patru
        sc.rel.lt.gam GAM1 GAM5 GLM1
        jumpnc GLM1 cinci
        sc.rel.lt.gam GAM1 GAM6 GLM1
        jumpnc GLM1 sase
        sc.rel.lt.gam GAM1 GAM7 GLM1
        jumpnc GLM1 sapte
        sc.rel.lt.gam GAM1 GAM8 GLM1
        jumpnc GLM1 opt
        jump est
doi:    sc.rel.lt.gam GAM2 GAM3 GLM1
        jumpnc GLM1 trei
        sc.rel.lt.gam GAM2 GAM4 GLM1
        jumpnc GLM1 patru
        sc.rel.lt.gam GAM2 GAM5 GLM1
        jumpnc GLM1 cinci
        sc.rel.lt.gam GAM2 GAM6 GLM1
        jumpnc GLM1 sase
        sc.rel.lt.gam GAM2 GAM7 GLM1
        jumpnc GLM1 sapte
```

```
        sc.rel.lt.gam GAM2 GAM8 GLM1
        jumpnc GLM1 opt
        jump vest
trei:   sc.rel.lt.gam GAM3 GAM4 GLM1
        jumpnc GLM1 patru
        sc.rel.lt.gam GAM3 GAM5 GLM1
        jumpnc GLM1 cinci
        sc.rel.lt.gam GAM3 GAM6 GLM1
        jumpnc GLM1 sase
        sc.rel.lt.gam GAM3 GAM7 GLM1
        jumpnc GLM1 sapte
        sc.rel.lt.gam GAM3 GAM8 GLM1
        jumpnc GLM1 opt
        jump nord
patru:  sc.rel.lt.gam GAM4 GAM5 GLM1
        jumpnc GLM1 cinci
        sc.rel.lt.gam GAM4 GAM6 GLM1
        jumpnc GLM1 sase
        sc.rel.lt.gam GAM4 GAM7 GLM1
        jumpnc GLM1 sapte
        sc.rel.lt.gam GAM4 GAM8 GLM1
        jumpnc GLM1 opt
        jump sud
cinci:  sc.rel.lt.gam GAM5 GAM6 GLM1
        jumpnc GLM1 sase
        sc.rel.lt.gam GAM5 GAM7 GLM1
        jumpnc GLM1 sapte
        sc.rel.lt.gam GAM5 GAM8 GLM1
        jumpnc GLM1 opt
        jump estnord
sase:   sc.rel.lt.gam GAM6 GAM7 GLM1
        jumpnc GLM1 sapte
        sc.rel.lt.gam GAM6 GAM8 GLM1
        jumpnc GLM1 opt
```

```
      jump vestnord
sapte: sc.rel.lt.gam GAM7 GAM8 GLM1
      jumpnc GLM1 opt
      jump estsud
opt:  jump vestsud
      ret
```

Anexa 6

```
; Program AMC pentru testarea algoritmului CNN
;de planificare a traiectoriei unui robot mobil,
;la deplasarea între doua puncte ale unui mediu cu
;obstacole, prin determinarea directiei optime
;la fiecare pas

start:
      host.load.tem explore.tem TEM1
      host.load.tem shifte.tem TEM2 ;shift est
      host.load.tem shiftv.tem TEM3 ;shift vest
      host.load.tem shiftn.tem TEM4 ;shift nord
      host.load.tem shifts.tem TEM5 ;shift sud
      host.load.tem shiften.tem TEM6;shift est-nord
      host.load.tem shiftvn.tem TEM7;shift vest-nord
      host.load.tem shiftes.tem TEM8;shift est-sud
      host.load.tem shiftvs.tem TEM9;shift vest-sud
      host.load.tem filwhite.tem TEM10
      ;imaginea mediului
      host.load.pic mediu.bmp LLM1 BIN
      ;imaginea tinta
      host.load.pic tinta.bmp LLM2 BIN
      ;imaginea start
      host.load.pic start.bmp LLM3 BIN
      ;imaginea traiectoriei planificate
      host.load.pic start.bmp LLM6 BIN
      ar.nand.llm LLM2 LLM2 LLM7
      ar.nand.llm LLM3 LLM3 LLM8
      host.display LLM1 1
      ar.nand.llm LLM1 LLM1 LLM1
      host.display LLM8 2
      host.display LLM7 3
```

```
;evaluarea distantei pozitiei actuale fata de pozitia tinta
      ar.tem TEM1 LLM1 LLM2 LAM3 200 1 1 LLM1 NIL
      host.display LAM3 4
loop:
      mov.llm.llm LLM3 LLM9
;Subprogram pentru evaluarea directiei optime
      call optdir
      ;se deplaseaza cu un pas pe directia aleasa
est:  ar.tem TEM2 LLM3 LLM3 LLM3 1 1 1 NIL NIL
      jump final
vest: ar.tem TEM3 LLM3 LLM3 LLM3 1 1 1 NIL NIL
      jump final
nord: ar.tem TEM4 LLM3 LLM3 LLM3 1 1 1 NIL NIL
      jump final
sud:  ar.tem TEM5 LLM3 LLM3 LLM3 1 1 1 NIL NIL
      jump final
estnord: ar.tem TEM6 LLM3 LLM3 LLM3 1 1 1 NIL NIL
      jump final
vestnord: ar.tem TEM7 LLM3 LLM3 LLM3 1 1 1 NIL NIL
      jump final
estsud: ar.tem TEM8 LLM3 LLM3 LLM3 1 1 1 NIL NIL
      jump final
vestsud: ar.tem TEM9 LLM3 LLM3 LLM3 1 1 1 NIL NIL
      jump final
final:
      ;atingerea tintei
      ar.xor.llm LLM3 LLM2 LLM5
      host.display LLM5 5
      conv.llm.lam LLM5 LAM4
      conv.lam.gam.avg LAM4 GAM11
      sc.rel.equ.gam -1 GAM11  GLM1
      ar.and.llm LLM6 LLM3 LLM6
      ar.nand.llm LLM6 LLM6 LLM10
      host.display LLM10 6
```

```
        jumpnc GLM1 loop
end:    end
;Subprogram pentru evaluarea directiei optime
optdir:
        ar.tem TEM2 LLM9 LLM9 LLM4 1 1 1 NIL NIL
        ar.tem TEM10 LLM4 LAM3 LAM2 1 1 1 LLM4 NIL
        conv.lam.gam.max LAM2 GAM1
        ar.tem TEM3 LLM9 LLM9 LLM4 1 1 1 NIL NIL
        ar.tem TEM10 LLM4 LAM3 LAM2 1 1 1 LLM4 NIL
        conv.lam.gam.max LAM2 GAM2
        ar.tem TEM4 LLM9 LLM9 LLM4 1 1 1 NIL NIL
        ar.tem TEM10 LLM4 LAM3 LAM2 1 1 1 LLM4 NIL
        conv.lam.gam.max LAM2 GAM3
        ar.tem TEM5 LLM9 LLM9 LLM4 1 1 1 NIL NIL
        ar.tem TEM10 LLM4 LAM3 LAM2 1 1 1 LLM4 NIL
        conv.lam.gam.max LAM2 GAM4
        ar.tem TEM6 LLM9 LLM9 LLM4 1 1 1 NIL NIL
        ar.tem TEM10 LLM4 LAM3 LAM2 1 -1 1 LLM4 NIL
        conv.lam.gam.max LAM2 GAM5
        ar.tem TEM7 LLM9 LLM9 LLM4 1 1 1 NIL NIL
        ar.tem TEM10 LLM4 LAM3 LAM2 1 1 1 LLM4 NIL
        conv.lam.gam.max LAM2 GAM6
        ar.tem TEM8 LLM9 LLM9 LLM4 1 1 1 NIL NIL
        ar.tem TEM10 LLM4 LAM3 LAM2 1 1 1 LLM4 NIL
        conv.lam.gam.max LAM2 GAM7
        ar.tem TEM9 LLM9 LLM9 LLM4 1 1 1 NIL NIL
        ar.tem TEM10 LLM4 LAM3 LAM2 1 1 1 LLM4 NIL
        conv.lam.gam.max LAM2 GAM8
        ;se determina directia optima
unu:    sc.rel.lt.gam GAM1 GAM2 GLM1
        jumpnc GLM1 doi
        sc.rel.lt.gam GAM1 GAM2 GLM1
        jumpnc GLM1 doi
        sc.rel.lt.gam GAM1 GAM3 GLM1
```

```
        jumpnc GLM1 trei
        sc.rel.lt.gam GAM1 GAM4 GLM1
        jumpnc GLM1 patru
        sc.rel.lt.gam GAM1 GAM5 GLM1
        jumpnc GLM1 cinci
        sc.rel.lt.gam GAM1 GAM6 GLM1
        jumpnc GLM1 sase
        sc.rel.lt.gam GAM1 GAM7 GLM1
        jumpnc GLM1 sapte
        sc.rel.lt.gam GAM1 GAM8 GLM1
        jumpnc GLM1 opt
        jump est
doi:    sc.rel.lt.gam GAM2 GAM3 GLM1
        jumpnc GLM1 trei
        sc.rel.lt.gam GAM2 GAM4 GLM1
        jumpnc GLM1 patru
        sc.rel.lt.gam GAM2 GAM5 GLM1
        jumpnc GLM1 cinci
        sc.rel.lt.gam GAM2 GAM6 GLM1
        jumpnc GLM1 sase
        sc.rel.lt.gam GAM2 GAM7 GLM1
        jumpnc GLM1 sapte
        sc.rel.lt.gam GAM2 GAM8 GLM1
        jumpnc GLM1 opt
        jump vest
trei:   sc.rel.lt.gam GAM3 GAM4 GLM1
        jumpnc GLM1 patru
        sc.rel.lt.gam GAM3 GAM5 GLM1
        jumpnc GLM1 cinci
        sc.rel.lt.gam GAM3 GAM6 GLM1
        jumpnc GLM1 sase
        sc.rel.lt.gam GAM3 GAM7 GLM1
        jumpnc GLM1 sapte
        sc.rel.lt.gam GAM3 GAM8 GLM1
```

```
      jumpnc GLM1 opt
      jump nord
patru: sc.rel.lt.gam GAM4 GAM5 GLM1
      jumpnc GLM1 cinci
      sc.rel.lt.gam GAM4 GAM6 GLM1
      jumpnc GLM1 sase
      sc.rel.lt.gam GAM4 GAM7 GLM1
      jumpnc GLM1 sapte
      sc.rel.lt.gam GAM4 GAM8 GLM1
      jumpnc GLM1 opt
      jump sud
cinci: sc.rel.lt.gam GAM5 GAM6 GLM1
      jumpnc GLM1 sase
      sc.rel.lt.gam GAM5 GAM7 GLM1
      jumpnc GLM1 sapte
      sc.rel.lt.gam GAM5 GAM8 GLM1
      jumpnc GLM1 opt
      jump estnord
sase: sc.rel.lt.gam GAM6 GAM7 GLM1
      jumpnc GLM1 sapte
      sc.rel.lt.gam GAM6 GAM8 GLM1
      jumpnc GLM1 opt
      jump vestnord
sapte: sc.rel.lt.gam GAM7 GAM8 GLM1
      jumpnc GLM1 opt
      jump estsud
opt:  jump vestsud
      ret
```

Bibliografie

[1] I.N. Aizemberg, N.N. Aizemberg, J. Vandewalle, "Precise edge detection: representation by Boolean functions, implementation on the CNN", Proceedings of IEEE International Workshop on Cellular Neural Networks and their Applications, (CNNA'98), pp. 301-306, London, 1998.

[2] P.K. Allen, A. Timcenko, B. Yoshimi, P. Michelman, "Automated tracking and grasping of a moving object with a robotic hand-eye system", IEEE Transactions on Robotics and Automation, (RA), Vol. 9, pp. 152-165, 1993.

[3] K. Bondor, T. Maghiar, S. Castrase, N.D. Trip, A. Gacsádi, "Digital measurements system for cryogenics temperatures that uses semiconductor diodes as sensors", Proceedings of the International Conference on New Trends of Signal Processing V, (NSSS 2000), Liptovsky Mikulás, pp. 215-220, 2000.

[4] A. Cimponeriu, "Rețele neuronale local liniare pentru comanda adaptivă a roboților cu reacție vizuală", Tezã de doctorat, Universitatea "Politehnica" Timişoara, 1997.

[5] B. Chandler, C. Rekeczky, Y. Nishio, A. Ushida, "Using adaptive simulated annealing in CNN template learning - a powerful alternative to genetic algorithms", Proceedings of European Conference on Circuit Theory and Design, (ECCTD'97), pp. 655-660, Budapest, 1997.

[6] C.C. Chang, K.T. Song, "Environment prediction for mobile robot in a dynamic environment", IEEE Transactions on Robotics and Automation, (RA), Vol.13, pp. 862-872, 1997.

[7] L.O. Chua, L. Yang, "Cellular neural networks: Theory", IEEE Transactions on Circuits and Systems, (CAS), Vol.35, pp. 1257-1272, 1988.

[8] L.O. Chua, L. Yang, "Cellular neural networks: Applications", IEEE Transactions on Circuits and Systems, (CAS), Vol.35, pp. 1273-1290, 1988.

[9] L.O. Chua, T. Roska, "Stability of a class of nonreciprocal cellular neural networks", IEEE Transactions on Circuits and Systems, (CAS), Vol.37, pp. 1520-1527, 1990.

[10] L.O. Chua, C.W. Wu, "On the universe of stable neural networks", International Journal of Circuit Theory and Applications, (CTA), Vol.20, pp. 497-517, 1992.

[11] L.O. Chua, T. Roska, "The CNN paradigm", IEEE Transactions on Circuits and Systems I: Fundamental Theory and Applications, (CAS-I), Vol.40, pp. 147-156, 1993.

[12] L.O. Chua, T. Roska, P.L. Venetianer, "The CNN is universal as the Turing machine", IEEE Transactions on Circuits and Systems I: Fundamental Theory and Applications, (CAS-I), Vol.40, pp. 289-291, 1993.

[13] L.O. Chua, M. Hasler, G.S. Moschytz, J. Neirynck, "Autonomous cellular neural network: A unified paradigm for pattern formation and active wave propagation", IEEE Transactions on Circuits and Systems I: Fundamental Theory and Applications, (CAS-I), Vol.42, pp. 559-577, 1995.

[14] L.O. Chua, T. Roska, "Cellular Neural Networks: Foundation and Primer", Version 1.7, Lecture Notes for the course EE129 at U.C. Berkeley, 1998.

[15] P.I. Corke, M.C. Good, "Dynamic effects in visual closed-loop systems", IEEE Transactions on Robotics and Automation, (RA), Vol. 12, pp. 671-683, 1996.

[16] K.R. Crounse, T. Roska, L.O. Chua, "Image halftoning with cellular neural networks", IEEE Transactions on Circuits and Systems II: Analog and Digital Signal Processing, (CAS-II), Vol.40, pp. 267-283, 1993.

[17] K.R. Crounse, L.O. Chua, "Methods for image processing and pattern formation in cellular neural networks: A tutorial", IEEE Transactions on Circuits and Systems I: Fundamental Theory and Applications, (CAS-I), Vol.42, pp. 583-601, 1995.

[18] K.R. Crounse, L.O. Chua, "The CNN universal machine is as universal as a Turing machine", IEEE Transactions on Circuits and Systems I: Fundamental Theory and Applications, (CAS-I), Vol.43, pp. 353-355, 1996.

[19] M. Csapodi, L. Nemes, G. Tóth, T. Roska, A. Radványi, "Some novel analogic CNN algorithms for object rotation, 3D interpolation-approximation, and a "door-in-a-floor" problem", Proceedings of the IEEE International Workshop on Cellular Neural Networks and their Applications, (CNNA'94), pp. 435-439, Rome, 1994.

[20] M. Csapodi, J. Vandewalle, B. Preneel, "CNN algorithms for video authentication and copyright protection", Journal of VLSI Signal Processing (JVSP Special Issue), pp. 449-463, Kluwer, 1999.

[21] R. Dominguez-Castro, S. Espejo, A. Rodriguez-Vazguez, R.A. Carmona, P. Földesy, Á. Zarándy, P. Szolgay, T. Szirányi, T. Roska, "0.8μm CMOS Two dimensional programmable mixed - signal focal - plane array processor with on - chip binary imaging and instructions storage", IEEE Journal of Solid State Circuits, Vol.32, pp. 1013-1026, 1997.

[22] D. Dumitrescu, H. Costin, "Rețele neuronale. Teorie şi aplicații", Editura Teora, Bucureşti, 1996.

[23] O.D. Faugeras, "Fundamentals in computer vision, an advanced course", Cambridge University Press, Cambridge, 1983.

[24] P. Földesy, L. Kék, T. Roska, Á. Zarândy, G. Bártfai, "Fault tolerant CNN template design and optimization based on chip measurements", Proceedings of IEEE International Workshop on Cellular Neural Networks and their Applications, (CNNA'98), pp. 404-409, London, 1998.

[25] A. Fujimori, P.N. Nikiforunk, M.M. Gupta, "Adaptive navigation of mobile robots with obstacle avoidance", IEEE Transactions on Robotics and Automation, (RA), Vol.13, pp. 596-602, 1997.

[26] A. Gacsádi, "Aspects regarding the optimization of visual information for an industrial robot". Proceedings of the International Conference on Renewable Sources and Environmental Electro-Technologies (RSEE'96), pp. 194-199, Băile Felix, 1996.

[27] A. Gacsádi, "Stadiul actual privind conducerea adaptivă a roboților industriali utilizând informația vizuală", Referat de doctorat-I, Universitatea "Politehnica" Timişoara, Facultatea de Electronică şi Telecomunicații, Timişoara, 1998.

[28] A. Gacsádi, "Utilizarea rețelelor neuronale la conducerea roboților industriali pe baza informației vizuale", Referat de doctorat-II, Universitatea "Politehnica" Timişoara, Facultatea de Electronică şi Telecomunicații, Timişoara, 1999.

[29] A. Gacsádi, C. Grava, "Simulation of the discrete time cellular neural network-DTCNN", Proceedings of the International Conference on Renewable Sources and Environmental Electro-Technologies (RSEE'98), pp. 99-104, Băile Felix, 1998.

[30] A. Gacsádi, T. Roska, P. Szolgay, "An analogic CNN algorithm for following continuously moving objects", Research report, Analogical and Neural Computing Laboratory, Computer and Automation Institute, Hungarian Academy of Sciences, (DNS-7-1999), Budapest 1999.

[31] A. Gacsádi, C. Grava, "Linear interpolation of 2D-images using cellular neural networks", Proceedings of the International Conference on New Trends of Signal Processing V, (NSSS 2000), pp. 178-183, Liptovsky Mikulás, 2000.

[32] A. Gacsádi, P. Szolgay, "An analogic CNN algorithm for following continuously moving objects", Proceedings of the IEEE International Workshop on Cellular Neural Networks and their Applications, (CNNA 2000), pp 99-105, Catania, 2000.

[33] A. Gacsádi, I. Turcaş, J. Vandewalle, "A CNN algorithm for determining the position of an object in an image", Technical Report, Katholieke Universiteit, ESAT-SISTA, Leuven, 2000.

[34] A. Gacsádi, I. Turcaş, J. Vandewalle, "Binary classification using CNN", Technical Report, Katholieke Universiteit, ESAT-SISTA, Leuven, 2000.

[35] A. Gacsádi, V. Tiponuț, "Path planning for a mobile robot in an environment with obstacles using cellular neural network", Proceedings of the International Conference on Renewable Sources and Environmental Electro-Technologies (RSEE 2000), in print, Oradea, 2000.

[36] A. Gacsádi, I. Turcaş, J. Vandewalle, "A CNN algorithm for determining the position of an object in an image", Proceedings of the International Conference on Renewable Sources and Environmental Electro-Technologies (RSEE 2000), in print, Oradea, 2000.

[37] A. Gacsádi, T. Roska, P. Szolgay, "Interpolation of 2D signals using CNN", Research report, Analogical and Neural Computing Laboratory, Computer and Automation Institute, Hungarian Academy of Sciences, (DNS-10-2000), Budapest, 2000.

[38] A. Gacsádi, V. Tiponuț, "Path planning for a mobile robot in an environment with obstacles using cellular neural network", Simpozion National cu participare internațională, (Robotica 2000), pp. 171-176, Oradea-Băile Felix, 2000.

[39] A. Gacsádi, P. Szolgay, "Interpolation of 2D signals using CNN", submitted to the European Conference on Circuit Theory and Design, (ECCTD'01), Espoo, 2001.

[40] M. Ghinea, V. Firițeanu, "MATLAB Calcul numeric-grafică-aplicații", Editura Teora, Bucureşti, 1995.

[41] J. Gomez-Ortega, E.F. Camacho, J. Quero, "Neural network local navigation of mobile robots in a moving obstacles environment", Proceedings of the Intelligent

Components and Instruments for Control Applications, (SICICA-94), pp. 263-267, Budapest, 1994.

[42] R.C. Gonzalez, R.E. Woods, "Digital image processing", Addison-Wesley Publishing Company, 1993.

[43] G. Grassi, G. Acciani, "Heteroassociative memories via globally asymptotically stable discrete-time cellular neural networks", Proceedings of the IEEE International Workshop on Cellular Neural Networks and their Applications, (CNNA 2000), pp. 141-145, Catania, 2000.

[44] C. Grava, A. Gacsádi, "Enlargement and interpolation of 2D signals using Cellular Neural Network", Proceedings of the International Conference on Engineering of Modern Electric Systems (EMES'99), pp. 183-188, Băile Felix, 1999.

[45] W.E.L. Grimson, "From images to surfaces - A Computational study of the human early visual system", The MIT Press, Cambridge/Massachusetts/London, 1981.

[46] W.E.L. Grimson, "An implementation of a computational theory of visual surface interpolation", Computer Vision, Graphics and Image Processing 22, pp. 39-69, 1983.

[47] H. Harrer, J. Nossek, "Discrete-time cellular neural networks", International Journal of Circuit Theory and Applications, (CTA), Vol. 20, pp. 453-467, 1992.

[48] H. Harrer, "Multiple-layer discrete-time cellular neural networks using time-variant templates", IEEE Transactions on Circuits and Systems II: Analog and Digital Signal Processing, (CAS-II), Vol.40, pp. 191-199, 1993.

[49] H. Harrer, P.L. Venetianer, J.A. Nossek, T. Roska, L.O. Chua, "Some examples of preprocessing analog images with discrete-time cellular neural networks", Proceedings of the IEEE International Workshop on Cellular Neural Networks and their Applications, (CNNA'94), pp. 201-206, Rome, 1994.

[50] K. Hashimoto, T. Ebine, H. Kimura, "Visual servoing with hand-eye manipulator-optimal control approach", IEEE Transactions on Robotics and Automation, (RA), Vol. 12, pp. 766-744, 1996.

[51] J.A. Hertz, A. Krogh, R.G. Palmer, "Introduction to the theory of neural computation", Addison-Wesley Publishing Company, 1991.

[52] G. Horváth, "Neurális hálózatok és műszaki alkalmazásai", Műegyetemi Kiadó, Budapest, 1998.

[53] H. Hu, M. Brady, "Dynamic global path planning with uncertainty for mobile robots in manufacturing", IEEE Transactions on Robotics and Automation, (RA), Vol.13, pp. 760-766, 1997.

[54] S. Hutchinson, G.D. Hager, P.I. Corke, "A tutorial on visual servo control", IEEE Transactions on Robotics and Automation, (RA), Vol. 12, pp. 651-670, 1996.

[55] A.K. Jain, "Fundamentals of digital image processing", Prentice-Hall Englewood Cliffs, 1989.

[56] M. Kanaya, M. Tanaka, "Robot multi-driving controls by cellular neural networks", Proceedings of the IEEE International Workshop on Cellular Neural Networks and their Applications, (CNNA'94), pp. 481-486, Rome, 1994.

[57] P. Keresztes, T. Bezák, Á. Zarándy, T. Roska, P. Szolgay, T. Hídvégi, P. Jónás, A. Katona, "Design methodology of an emulated digital CNNUM chip", Proceedings of the International Conference on Engineering of Modern Electric Systems (EMES'99), pp. 208-216, Băile Felix, 1999.

[58] R. Kelly, "Robust asymptotically stable visual servoing of planar robots", IEEE Transactions on Robotics and Automation, (RA), Vol. 12, pp. 759-766, 1996.

[59] L. Kék, Á. Zarándy, "Implementation of large neighborhood nonlinear templates on the CNN Universal Machine", International Journal of Circuit Theory and Applications - Special Issue: Theory, Design and Applications of Cellular Neural Networks: Part I: Theory, (CTA Special Issue - I), Vol.26, pp. 551-566, 1998.

[60] Z. Kis, P. Szolgay, "Real-time adaptive control of robot manipulators using cellular neural networks", Proceedings of European Conference on Circuit Theory and Design, (ECCTD'97), pp. 628-633, Budapest, 1997.

[61] T. Kozek, T. Roska, L. O. Chua, "Genetic algorithm for CNN template learning", IEEE Transactions on Circuits and Systems I: Fundamental Theory and Applications, (CAS-I), Vol.35, pp. 392-402, 1993.

[62] T. Kozek, T. Roska, "A double time-scale CNN for solving two-dimensional Navier-Stokes equations", International Journal of Circuit Theory and Applications, (CTA), Vol 24, pp. 49-55, 1996.

[63] T. Kozek, C.W. Wu, Á. Zarándy, H. Chen, T. Roska, M. Kunt, L.O. Chua, "New results and measurements related to some tasks in object-oriented dynamic image coding using CNN universal chip", IEEE Transactions on Circuits and Systems for Video Technology, Vol.7, pp. 606-613, 1997.

[64] P. Kostic, B. Reljin, I. Reljin, "Cellular neural network for trajectory tracking and predicting", Proceedings of European Conference on Circuit Theory and Design, (ECCTD'99), pp. 956-959, Stresa, 1999.

[65] J.C. Latombe, "Robot motion planning". Kluwer Academic Publishers, Boston/Dordrecht/London, 1991.

[66] K. László, F. Ziliani, T. Roska, M. Kunt, "Early segmentation in video compression using CNN processors", Proceedings of IEEE International Workshop on Cellular Neural Networks and their Applications, (CNNA'98), pp. 175-180, London, 1998.

[67] F.L. Lewis, K. Liu, A. Yesildirek, "Neural net robot controller with guaranteed tracking performance", IEEE Transactions on Neural networks, Vol. 6, pp. 703-715, 1995.

[68] G. Linan, S. Espejo, R. Dominguez-Castro, E. Roca, A. Rodriguez-Vazquez, "CNNUC3: A mixed-signal 64x64 CNN Universal Chip", Proceedings of the International Conference on Microelectronics for Neural, Fuzzy and Bio-inspired Systems, (MicroNeuro99), pp. 61-68, Granada, 1999.

[69] F. Luthon, D. Dragomirescu, "A Cellular analog network for MRF-based video motion detection", IEEE Transactions on Circuits and Systems I: Special Issue on Bio-Inspired Processors and Cellular Neural Networks for Vision, (CAS-I Special Issue), Vol. 46, pp. 281-293, 1999.

[70] E. Malis, F. Chaumette, S. Boudet, "2-1/2-D visual servoing", IEEE Transactions on Robotics and Automation, (RA), Vol. 15, pp. 238-250, 1999.

[71] T. Matsumoto, T. Yokohama, H. Suzuki, R. Furukawa, A. Oshimito, T. Shimmi, Y. Matsushita, T. Seo, L.O. Chua, "Several image processing examples by CNN", Proceedings of IEEE International Workshop on Cellular Neural Networks and their Applications, (CNNA'90), pp. 100-111, Budapest, 1990.

[72] M. Meng, S.X. Yang, "A neural network approach to real-time trajectory generation", Proceedings of the IEEE International Conference on Robotics and Automation, pp. 1725-1730, Leuven, 1998.

[73] W.T. Miller, R.P. Hewes, F.H. Glanz, L.G. Kraft, "Real-time dynamic control of an industrial manipulator using a neural-network-based learning controller", IEEE Transactions on Robotics and Automation, (RA), Vol. 6, pp. 1-9, 1990.

[74] B. Mirzai, D. Lim, G.S. Moschytz, "Applications of CNN processing by template decomposition", Proceedings of IEEE International Workshop on Cellular Neural Networks and their Applications, (CNNA'98), pp. 379-384, London, 1998.

[75] B.J. Nelson, P.K. Khsola, "Force and vision resolvability for assimilating disparate sensory feedback", IEEE Transactions on Robotics and Automation, (RA), Vol. 12, pp. 714-731, 1996.

[76] J.A. Nossek, G. Seiler, T. Roska, L.O. Chua, "Cellular neural networks: Theory and circuit design", International Journal of Circuit Theory and Applications, (CTA), Vol. 20, pp. 533-553, 1992.

[77] J.A. Nossek, "Design and learning with cellular neural networks", Proceedings of IEEE International Workshop on Cellular Neural Networks and their Applications, (CNNA'94), pp.137-146, Rome, 1994.

[78] A. Paasio, A. Kananen, K. Halonen, V. Porra, "A 48 by 48 CNN chip operating with B/W images", Proceedings of IEEE International Conference on Electronics, Circuits and systems, (ICECS'98), pp. 191-194, Lisboa, 1998.

[79] A. Paasio, A. Kananen, V. Porra, "A 176 x 144 processor binary I/O CNN-UM chip design", Proceedings of European Conference on Circuit Theory and Design, Design Automation Day, (ECCTD'99-DAD), pp. 941-944, Stresa, 1999.

[80] N.P. Papanikolopoulus, P.K. Khosla, T. Kanade, "Visual tracking of a moving target by a camera mounted on a robot: a combination of control and vision", IEEE Transactions on Robotics and Automation, (RA), Vol. 9, pp. 14-35, 1993.

[81] P.R. Paul, "Robot manipulators: mathematics, programming, and control. The computer control of robot manipulators", MIT Press, Cambridge/Massachusetts/ London, 1982.

[82] D. Pomerleau, "A neural network perception for mobile robot guidance", Kluwer Academic Publishers, Boston/Dordrecht/London, 1993.

[83] E. Pop, I. Naforniță, V. Tiponuț, A. Mihăescu, L. Toma, " Metode în prelucrarea numerică a semnalelor", Editura Facla, Timişoara, Vol. I-1986, Vol. II-1989.

[84] A. Radványi, T. Kozek, L.O. Chua, "A CNN solution for Depth estimation from binocular stereo imagery", Proceedings of IEEE International Workshop on Cellular Neural Networks and their Applications, (CNNA'98), pp. 218-223, London, 1998.

[85] C. Rekeczky, "MATCNN-Analogic CNN Simulation toolbox for MATLAB", Version 1.0, Analogical and Neural Computing Laboratory, Computer and Automation Institute, Hungarian Academy of Science, Budapest, 1997.

[86] C. Rekeczky, "Active contour and skeleton models in continuous-time CNN", Proceedings of European Conference on Circuit Theory and Design, (ECCTD'99), pp. 1015-1018, Stresa, 1999.

[87] T. Roska, L.O. Chua, "The CNN Universal Machine: An analogic array computer", IEEE Transactions on Circuits and Systems II: Analog and Digital Signal Processing, (CAS-II), Vol.40, pp. 163-173, 1993.

[88] T. Roska, L.O. Chua, "Cellular neural networks with non-linear and delay-type template elements and non-uniform grids", International Journal of Circuit Theory and Applications, (CTA), Vol. 20, pp. 469-481, 1992.

[89] T. Roska, T. Boros, A. Radványi, "Detecting moving and standing objects using cellular neural networks", International Journal of Circuit Theory and Applications, (CTA), Vol. 20, pp. 613-628, 1992.

[90] T. Roska, J. Hámori, E. Lábos, K. Lotz, L. Orzó, J. Takács, P.L. Venetianer, Z. Vidnyánszky, Á. Zarándy, "The use of CNN models in the subcortical visual pathway", IEEE Transactions on Circuits and Systems I: Fundamental Theory and Applications, (CAS-I), Vol.40, pp.182-195, 1993.

[91] T. Roska, L.O. Chua, D. Wolf, T. Kozek, R. Tetzlaff, F. Puffer, "Simulating nonliniar waves and partial differential equations via CNN-part I: Basic techniques", IEEE Transactions on Circuits and Systems I: Fundamental Theory and Applications, (CAS-I), Vol.42, pp. 807-815, 1995.

[92] T. Roska, "Analogic CNN computing: architectural, implementation, and algorithmic advances - a review", Proceedings of IEEE International Workshop on Cellular Neural Networks and their Applications, (CNNA'98), pp. 3-10, London, 1998.

[93] A. Schultz, C. Rekeczky, I. Szatmári, T. Roska, L.O. Chua, "Spatio-temporal CNN algorithm for object segmentation and object recognition", Proceedings of IEEE International Workshop on Cellular Neural Networks and their Applications, (CNNA'98), pp. 347-352, London, 1998.

[94] B. Siemiątkowska, "Cellular neural network for mobile robot navigation", Proceedings of the IEEE International Workshop on Cellular Neural Networks and their Applications, (CNNA'94), pp. 285-290, Rome, 1994.

[95] B.E. Shi, "Gabor-type filtering in space and time with cellular neural networks", IEEE Transactions on Circuits and Systems I: Fundamental Theory and Applications, (CAS-I), Vol.45, pp. 121-132, 1998.

[96] I. Simionescu, M. Dranga, V. Moise, "Metode numerice în tehnică", Editura Tehnică, Bucureşti, 1995.

[97] K. Slot, "Determination of cellular neural networks parameters for feature detection of two-dimensional images", Proceedings of IEEE International Workshop on Cellular Neural Networks and their Applications, (CNNA'90), pp. 73-81, Budapest, 1990.

[98] A. Stoffels, T. Roska, L.O. Chua, "An object-oriented approach to video coding via the CNN Universal Machine", Proceedings of IEEE International Workshop

on Cellular Neural Networks and their Applications, (CNNA'96), pp. 13-18, Sevilla, 1996.

[99] P. Szolgay, A. Katona, Á. Kiss, L. Székely, A. Veress, "An optical robot path tracking system", Toward the Visual Microprocessor - VLSI Design and Use of Cellular Network Universale Machines (Toward the Visual Microprocessor), John Wiley and Sons, 1997.

[100] P. Szolgay, K. Tömördi, "Analogic algorithms for optical detection of breaks and short circuits on the layouts of printed circuits boards using CNN", International Journal of Circuit Theory and Applications - Special Issue: Theory, Design and Applications of Cellular Neural Networks: Part II: Design and Applications, (CTA Special Issue - II), Vol.27, pp. 103-116, 1999.

[101] S. Tan, J. Hao, J. Vandewalle, "Cellular neural networks as a model of associative memories", Proceedings of IEEE International Workshop on Cellular Neural Networks and their Applications, (CNNA'90), pp. 26-35, Budapest, 1990.

[102] D. Terzopoulos, "Multilevel computational processes for visual surface reconstruction", Computer Vision, Graphics and Image Processing 24, pp. 52-95, 1983.

[103] R. Tetzlaff, R. Kunz, C. Ames, D. Wolf, "Analysis of brain electrical activity in epilepsy with cellular neural network", Proceedings of European Conference on Circuit Theory and Design, (ECCTD'99), pp. 1007-1010, Stresa, 1999.

[104] V. Tiponuț "Rețele neuronale", Note de curs, Universitatea "Politehnica" Timişoara, Facultatea de Electronică şi Telecomunicații, Timişoara, 1994.

[105] G. Toderan, M. Coşteiu, M. Giurgiu, "Rețele neuronale", Editura Microinformatica, Cluj-Napoca, 1994.

[106] P.G. Tzionas, A. Thanailakis, P.G. Tsalides, "Collision-free path planning for a diamond-shaped robot using two-dimensional cellular automata", IEEE Transactions on Robotics and Automation, (RA), Vol.13, pp. 237-250, 1997.

[107] C. Venaille, G. Wells, C. Torras, "Application of neural networks to image-based control of robot arms", Proceedings of the Intelligent Components and Instruments for Control Applications, (SICICA-94), pp. 281-285, Budapest, 1994.

[108] P.L. Venetianer, F. Werblin, T. Roska, L.O. Chua, "Analogic CNN algorithms for some image compression and restoration tasks", IEEE Transactions on Circuits and Systems I: Fundamental Theory and Applications, (CAS-I), Vol.42, pp. 278-284, 1995.

[109] P.L. Venetianer, P. Szolgay, K.R. Crounse, T. Roska, L.O. Chua, "Analogue combinatorics and cellular automata - key algorithms and layout design", International Journal of Circuit Theory and Applications, (CTA), Vol. 24, pp. 145-164, 1996.

[110] L.E. Weiss, A.C. Sanderson, C.P. Neuman, "Dynamic sensor-based control of robots with visual feedback", IEEE Transactions on Robotics and Automation, (RA), Vol. 3, pp. 404-417, 1987.

[111] F. Werblin, T. Roska, L.O. Chua, "The analogic cellular neural network as a bionic eye", International Journal of Circuit Theory and Applications, (CTA), Vol.23, pp. 541-569, 1995.

[112] W.J. Wilson, C.C. Williams Hulls, G.S. Bell, "Relative end-effector control using cartesian position based visual servoing", IEEE Transactions on Robotics and Automation, (RA), Vol. 12, pp. 684-696, 1996.

[113] L.B. Yang, T. Yang, B.S. Chen, "Moving point target detection using cellular neural networks", Proceedings of the IEEE International Workshop on Cellular Neural Networks and their Applications, (CNNA'94), pp. 445-450, Rome, 1994.

[114] X. Yang, T. Yang, L.B. Yang, "Extracting focused object from defocused background using cellular neural networks", Proceedings of the IEEE International Workshop on Cellular Neural Networks and their Applications, (CNNA'94), pp. 451-456, Rome, 1994.

[115] S.X. Yang, M. Meng, "An efficient neural network approach to dynamic robot motion planning", Neural Networks, Vol. 13, pp. 143-148, 2000.

[116] A. Zanela, S.Taraglio, "A Robustness study of a CNN based stereo vision algorithm", Proceedings of IEEE International Workshop on Cellular Neural Networks and their Applications, (CNNA'98), pp. 325-330, London, 1998.

[117] Á. Zarándy, F. Werblin, T. Roska, L.O. Chua, "Novel types of analogic CNN algorithms for recognizing bank-notes", Proceedings of the IEEE International Workshop on Cellular Neural Networks and their Applications, (CNNA'94), pp. 273-278, Rome, 1994.

[118] Á. Zarándy, T. Roska, "CNN template design strategies and fault tolerant CNN template design - a survey", Proceedings European Conference on Circuit Theory and Design, Design Automation Day, (ECCTD'97-DAD), pp. 178-197, Budapest, 1997.

[119] Á. Zarándy, E. Grawes, T. Roska, F. Werblin, L.O. Chua, "CNN model for identifying colors under different illumination conditions via Land's experiments", Proceedings of IEEE International Workshop on Cellular Neural Networks and their Applications, (CNNA'96), pp. 163-168, Seville, 1996.

[120] Á. Zarándy, A. Stoffels, T. Roska, L.O. Chua, "Morphological operators on the CNN Universal Machine", Proceedings of IEEE International Workshop on Cellular Neural Networks and their Applications, (CNNA'96), pp. 151-156, Seville, 1996.

[121] Á. Zarándy, T. Roska, F. Werblin, L.O. Chua, "Intelligent image resolution enhancement by the CNN Universal Machine and its relevance to TV picture enhancement", Proceedings of the International Symposium on Nonlinear Theory and Applications, (NOLTA'95), pp.701-706, Las Vegas, 1995.

[122] H. Zhuang, K. Wang, Z.S. Roth, "Simultaneous calibration of a robot and a hand-mounted camera", IEEE Transactions on Robotics and Automation, (RA), Vol. 11, pp. 649-660, 1995.

[123] *** "CadetWin-99, CNN application development environment and toolkit under Windows" Version 3.0, Analogical and Neural Computing Laboratory, Computer and Automation Institute, Hungarian Academy of Science, Budapest, 1999.

[124] *** "CadetWin-99, VisMouse-SimCNN, Visual mouse platform and multi-layer CNN simulator", User's guide, Version 3.0, Analogical and Neural Computing Laboratory, Computer and Automation Institute, Hungarian Academy of Sciences, Budapest, 1999.

[125] *** "C S L-CNN Software Library (Templates and Algorithms)", Version 7.3, Analogical and Neural Computing Laboratory, Computer and Automation Institute, Hungarian Academy of Sciences, Budapest, 1999.

[126] *** "CadetWin-99, TemMaster, Template design and optimization tool for binary input-output CNNs", User's guide, Version 3.0, Analogical and Neural Computing Laboratory, Computer and Automation Institute, Hungarian Academy of Sciences, Budapest, 1999.

[127] *** "CadetWin-99, VisMouse-CNN visual mouse software platform for Windows", Reference manual, Version 3.0, Analogical and Neural Computing Laboratory, Computer and Automation Institute, Hungarian Academy of Sciences, Budapest, 1999.

[128] *** "CadetWin-99, SimCNN-Multi-layer CNN simulator for visual mouse platform", Reference Manual, Version 3.0, Analogical and Neural Computing Laboratory, Computer and Automation Institute, Hungarian Academy of Sciences, Budapest, 1999.

[129] *** "CadetWin-99, CNN Alpha language and compiler", Reference Manual, Version 3.0, Analogical and Neural Computing Laboratory, Computer and Automation Institute, Hungarian Academy of Sciences, Budapest, 1999.

[130] *** "CadetwWin-99, Extended analogic macro code (AMC) and interpreter", Reference Manual, Version 3.0, Analogical and Neural Computing Laboratory, Computer and Automation Institute, Hungarian Academy of Sciences, Budapest, 1999.

[131] *** "CCPS, CNN chip prototyping system", User's Guide, Analogical and Neural Computing Laboratory, Computer and Automation Institute, Hungarian Academy of Sciences, Budapest, 1997.

[132] *** EVI-D31 Sony, Intelligent communications color video camera, User's Guide, 1997.