CONTRIBUŢII LA UTILIZAREA PRELUCRĂRII AUTOMATE A IMAGINII LA ROBOŢI DE MANIPULARE

Teză destinată obţinerii
titlului ştiinţific de doctor inginer
la
Universitatea „Politehnica” din Timişoara
în domeniul INGINERIE MECANICĂ
de către

Ing. Cristian Pop

Conducător ştiinţific:	Prof.univ.dr.ing. Arjana Davidescu
Referenţi ştiinţifici:	Prof.univ.dr.ing. Doina PÎSLĂ Prof.univ.dr.ing. Ioan DOROFTEI Prof.univ.dr.ing. Valer DOLGA

Ziua susţinerii tezei: 17.01.2013

Seriile Teze de doctorat ale UPT sunt:

1. Automatică
2. Chimie
3. Energetică
4. Ingineria Chimică
5. Inginerie Civilă
6. Inginerie Electrică
7. Inginerie Electronică şi Telecomunicaţii
8. Inginerie Industrială
9. Inginerie Mecanică
10. Ştiinţa Calculatoarelor
11. Ştiinţa şi Ingineria Materialelor

Universitatea „Politehnica” din Timişoara a iniţiat seriile de mai sus în scopul diseminării expertizei, cunoştinţelor şi rezultatelor cercetărilor întreprinse în cadrul şcolii doctorale a universităţii. Seriile conţin, potrivit H.B.Ex.S Nr. 14 / 14.07.2006, tezele de doctorat susţinute în universitate începând cu 1 octombrie 2006.

România, 300159 Timişoara, Bd. Republicii 9,
tel. 0256 403823, fax. 0256 403221
e-mail: editura@edipol.upt.ro

Cuvânt înainte

Teza de doctorat a fost elaborată pe parcursul activităţii mele prestate în perioada stagiului ca doctorand cu frecvenţă în cadrul Departamentului de Mecatronică, Facultatea de Mecanică, Universitatea ”Politehnica” din Timişoara.

Mulţumiri deosebite se cuvin conducătorului de doctorat prof. dr. ing. Arjana Davidescu, pentru sfaturile şi suportul acordat în toate problemele apărute pe parcursul celor trei ani de doctorat.

Doresc să mulţumesc d-nei şef. lucr. dr. ing. Sanda Margareta Grigorescu, pentru sprijinul acordat în rezolvarea problemelor care au apărut, în mod inevitabil, pe parcursul implementării pe roboţi a aplicaţiilor prezentate în teza de faţă.

De asemenea, mulţumesc d-lui prof. dr. ing. Valer Dolga şi d-lui prof. dr. ing. George Savii pentru indicaţiile utile finalizării tezei de doctorat.

Mulţumesc domnului profesor Alessandro Gasparetto pentru suportul acordat pe durata desfăşurării stagiului de cercetare în străinătate, în cadrul Universităţii de Studii din Udine, Italia.

Aş dori să mulţumesc tuturor cadrelor didactice din cadrul Departamentului de Mecatronică, care mi-au asigurat un mediu adecvat pentru cercetare.

Îmi exprim întreaga consideraţie faţă de membrii comisiei de doctorat, care au răspuns solicitării de a face parte din comisia de analiză a tezei, pentru observaţiile făcute şi pentru timpul acordat lucrării.

De asemenea mulţumesc tuturor celor care m-au ajutat cu diverse sfaturi şi indicaţii utile finalizării tezei de doctorat, precum şi managementului Proiectului de burse doctorale nr. 88/1.5/S/50783.

Nu în ultimul rând, doresc să adresez mulţumiri familiei pentru înţelegerea şi suportul moral pe care mi le-au oferit pe întreaga perioadă a desfăşurării studiilor doctorale şi nu numai.

Timişoara, ianuarie 2013 Ing. Cristian Pop

Teza de doctorat a fost realizată cu sprijin parţial din grantul strategic POSDRU/88/1.5/S/50783, cofinanţat din Fondul Social European "Investeste în oameni", în cadrul Programului Operaţional Sectorial Dezvoltare Resurse Umane 2007-2013.

Pop,Cristian

Contribuţii la utilizarea prelucrării automate a imaginii la roboţi de manipulare

Teze de doctorat ale UPT, Seria 9, Nr. 128, Editura Politehnica, 2012, 187 pagini, 87 figuri, 18 tabele.

ISSN: 1842-4937

ISBN: 978-606-554-594-6

Cuvinte cheie:
Prelucrare de imagini, Roboţi de manipulare, Vedere artificială, Vedere robotizată, Sisteme Servoing, Recunoaștere obiecte, Calibrare camere video.

Rezumat,
Teza de faţă propune o serie de contribuţii, atât teoretice cât şi practice, la utilizarea prelucrării automate a imaginii pentru conducerea unor roboţi industriali. Aceasta abordează o temă de cercetare multidisciplinară, ce implică cunoştiinţe din domeniul ingineriei mecanice, robotice şi ştiinţa calculatoarelor.
Lucrarea cuprinde aspecte teoretice, cu privire la principalele componente ale unui sistem de vedere robotizat, aplicaţii actuale de vedere robotizată din industrie, şi o analiză a metodelor de calibrare a camerelor video, existente în literatura de specialitate, finalizată prin iplementarea a două metode în mediul de lucru Matlab.
Aplicaţiile ce au fost dezvoltate, cu grad de dificultate gradual, presupun anumiţi algoritmi de vedere computerizată, concepuţi şi implementaţi de autor, pentru conducerea autonomă şi semi-autonomă a roboţilor, cu scopul de îmbunătăţire a proceselor de manipulare robotizate. Aceşti algoritmi s-au axat în principiu pe operaţii de detectare, recunoaştere şi identificare a unor obiecte variate, în vederea extragerii anumitor caracteristici necesare operaţiei de conducere a roboţilor.
Este prezentat, de asemenea, şi un model matematic original al autorului, ce permite determinarea poziţiei unor obiecte în spaţiu, pe baza coordonatelor a patru puncte caracteristice, extrase din două imagini ale obiectului, ce sunt prelevate de două camere situate în planuri perpendiculare.

Cuprins

Lista de tabele

Lista de figuri

1. INTRODUCERE

1.1. Generalităţi

Vederea este sistemul senzorial esenţial, ce furnizează, informaţia calitativă şi cantitativă necesară pentru a înţelege şi percepe lumea înconjurătoare, pentru a interacţiona cu mediul, cu scopul de a realiza acţiuni de manipulare a diverselor obiecte existente în jur. Din acest motiv, mulţi cercetători, au încercat de-a lungul timpului să creeze şi să implementeze sisteme de vedere artificială pe diverse echipamente sau maşini. Printre aceste echipamente se numără sistemele robotizate, care s-au dezvoltat în paralel cu tehnica şi tehnologia informaţiei şi au ajuns în prezent la o mare diversitate clasificându-se în trei mari categorii:

1. Roboţi industriali care apar undeva la începutul anilor 1960 şi care se caracterizează prin existenţa unui mediu structurat cu obiecte situate în puncte de precizie şi sarcini predefinite prin program;
2. Roboţi de prestări servicii (1984);
3. Roboţi personali (2003).

Tema abordată de lucrarea de faţă se referă doar la prima categorie de roboţi. Robotul industrial, conform standardului 8373 din 1994 al Institutului de Standardizare Internaţional (ISO) se defineşte ca „un manipulator controlat automat, reprogramabil, multifuncţional, programabil pe trei sau mai multe axe, care poate fi fix sau mobil şi este utilizat în aplicaţii de automatizări industriale” (Austin 2005), (Mathia 2010).

Structura şi sistemul senzorial al roboţilor industriali este asemănător cu cel al organismului uman. Senzorii cu care sunt dotaţi în general aceştia se clasifică în două categorii:

1. Senzorii interni care au rolul de a determina starea curentă a unui modul şi anume în speţă poziţiile/deplasărie din cuplele cinematice;
2. Senzorii externi care prelevează informaţii din mediul înconjurător, precum senzorii de proximitate sau de forţă, ;

Cei mai importanţi senzori externi cu care poate fi dotat un robot industrial modern, inteligent, care este capabil să analizeze modificările ce pot să apară în mediul inconjurător şi să acţioneze în consecinţă, sunt senzorii de forţă şi cei vizuali.

Vederea artificială reprezintă sistemul senzorial care îmbunătăţeşte în cea mai mare măsură performanţele unui robot. Pentru ca robotul industrial să poată interacţiona cu diferite medii de lucru fără a exista un contact direct, respectiv pentru ca să se poată adapta diferitelor medii de lucru şi diferiţilor factori perturbatori ce pot să intervină pe parcursul activităţii sale, acesta trebuie să fie înzestrat cu un sistem de vedere artificială care să permită prelevarea informaţiilor vizuale.

Elementul senzorial al acestui sistem este unul pasiv ce achiziţionează informaţia din spaţiul de lucru într-un mod non-invaziv. În momentul de faţă aceşti senzori se găsesc într-o gamă diversificată, sunt destul de performanţi din punct de vedere al calităţii şi cantităţii informaţiei prelevate, şi au un preţ de cost redus, fapt ce face ca aceştia să fie utilizaţi tot mai des în operaţiile pe care roboţii le pot realiza.

Aceste sisteme de vedere artificială presupun următoarele îmbunătățiri față de sistemul de vedere uman:

- Pot utiliza rezoluții foarte mari;
- Pot utiliza întreg spectrul electromagnetic;
- Pot identifica fiecare culoare în parte;
- Pot diminua timpul de răspuns;
- Pot prelua imagini din diverse zone unde omul nu are acces.

Informaţiile vizuale prelevate de către aceste sisteme de vedere artificială, pentru a putea fi folosite de către robot în scopul navigării şi interacţionării lui cu mediul înconjurător, respectiv pentru a putea realiza diverse operaţii sau aplicații, trebuie să fie în prealabil procesate. Metodele şi tehnicile ce ţin de această operaţie aparţin domeniului vederii computerizate şi ele presupun prelucrarea, analiza şi înţelegerea informaţiilor prelevate din lumea reală sub forma de imagini şi transformarea lor în informaţii sub formă numerică sau simbolică ce pot fi ulterior utilizate de către sistemul de control robotului.

Avantajele pe care le presupune integrarea unui sistem de vedere artificială într-un sistem robotizat şi implementarea algoritmilor specifici de vedere computerizată sunt:

- Productivitate crescută;
- Diminuarea sau compensarea erorilor;
- Precizie ridicată;
- Creşterea calității produselor;
- Reducerea costului produselor.

Pe lângă acestea, un robot prevăzut cu un sistem de vedere artificial şi algoritmi de vedere computerizată poate realiza în mod semi-automat sau automat operaţii mai complexe în comparaţie cu un robot clasic și poate, de asemenea, lucra în medii periculoase pentru om fără ca performanţele acestuia să scadă.

1.2. Sisteme servoing vizuale

„Vedere Robotizată” sau „Viziunea Robotizată” (Robot Vision) se ocupă cu integrarea echipamentelor şi dispozitivelor unui sistem de vedere artificial (artificial vision system) şi implementarea unor algoritmi de vedere computerizată (computer vision) într-un sistem robotizat, astfel încât acesta să fie capabil să realizeze într-un mod cât mai flexibil diverse aplicaţii într-un mediu structurat sau nestructurat. Acest domeniu reprezintă, practic, o ramificaţie în zona roboticii a sferei de cercetare care se ocupă cu înzestrarea tuturor maşinilor şi sistemelor automate cu sisteme de vedere artificială (Machine Vision). Ariile de cercetare înrudite cu care acest domeniu se întrepătrunde sunt ilustrate schematic în figura 1.1 (Ude, 2010).

Subdomeniul viziunii robotizate care se ocupă în mod concret doar cu cercetarea, proiectarea, dezvoltarea sistemelor, metodelor/tehnicilor şi aplicațiilor de control al roboţilor pe baza informaţiei obţinute de la senzorii vizuali poartă numele de „Controlul Roboților pe Baza Vederii Artificiale” sau „Servoing Vizual” (Vision - Based Robot Control sau Visual Servoing - VS).

Scopul acestui subdomeniu este de a utiliza informația vizuală prelevată de la un senzor optic sau un sistem de senzori optici amplasat în mediul de lucru sau pe robot, pentru a controla poziţia şi orientarea robotului în raport cu un sistem de referinţă ataşat mediului sau în raport cu obiecte ce se află în mediu.

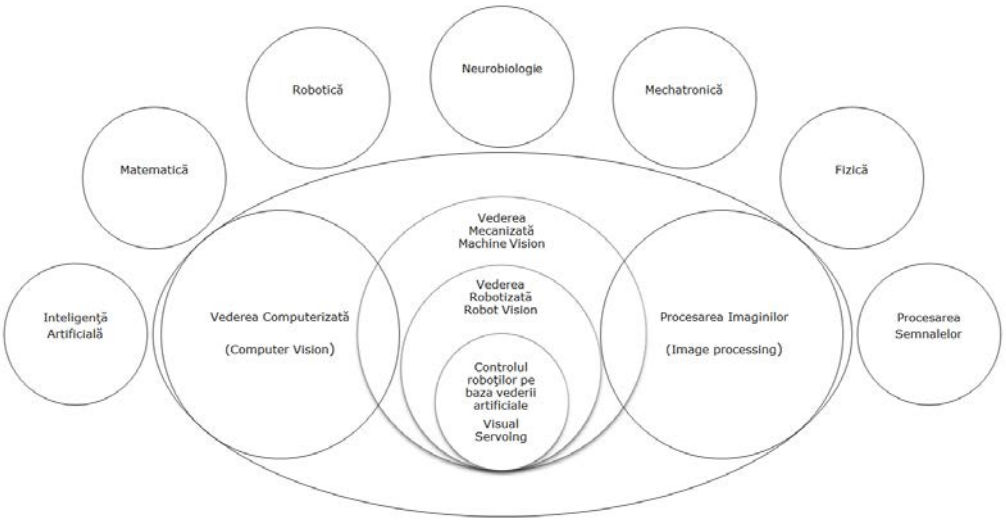


Figura 1. 1 Amplasarea domeniului de cercetare în raport cu alte domenii de activitate.

Sistemele servoing vizuale (VS) sunt prezentate în literatura de specialitate într-o mare varietate de forme (Tell, 2002), ceea ce face ca pentru diferențierea lor, aceste sisteme să fie clasificate pe baza unor caracteristici în următoarele categorii:

1. Din punct de vedere al configurației senzorilor optici (video) figura 1.2:
   a. Sisteme servoing vizuale cu configuraţie „ochi pe mână” (eye – in – hand systems). Aceste sisteme se caracterizează prin faptul că una sau mai multe camere video sunt prinse fizic de robot. În cazul unui robot industrial, în general, sunt ataşate de efectorul final al acestuia, dar ele pot fi de asemenea prinse şi de un alt element al robotului;
   b. Sisteme servoing vizuale cu configuraţie „ochi la mână” (eye – to – hand systems), se definesc prin camere amplasate arbitrar în spaţiul de lucru. Camerele vizualizează în permanenţă obiectul ţintă, un obiect fix sau ambele (Allen, 1993);
   c. Sisteme servoing vizuale cu configuraţie „mixtă” (multiple on/off board cameras systems) sunt sisteme în care camerele video sunt ataşate atât pe robot cât şi în spaţiu de lucru (Kragic, 2003).

Prima variantă (eye-in-hand) are avantajul unei bune precizii de determinare a poziției şi orientării obiectului ţintă în spaţiul de lucru. Acesta lucru este posibil datorită creşterii rezoluţiei obiectului în imagine în urma apropierii camerei video de ţintă. Aceasta variantă reprezintă abordarea cea mai comună, care permite robotului o libertate ridicată. Pentru implementare este necesar să se realizeze o singură dată calibrarea sistemului, respectiv determinarea matricei de transformare dintre efectorul final al robotului şi camera video datorită faptului că cele două componente sunt strict conectate, respectiv relaţia dintre acestea este definită şi rămâne neschimbată (Corke, 1996), (Vona, 2010), (Copot, 2011).

Cea de-a doua variantă (eye-to-hand) este utilizată mai rar deoarece prezintă un grad de complexitate mai ridicat. Acest lucru se datorează operaţiei de

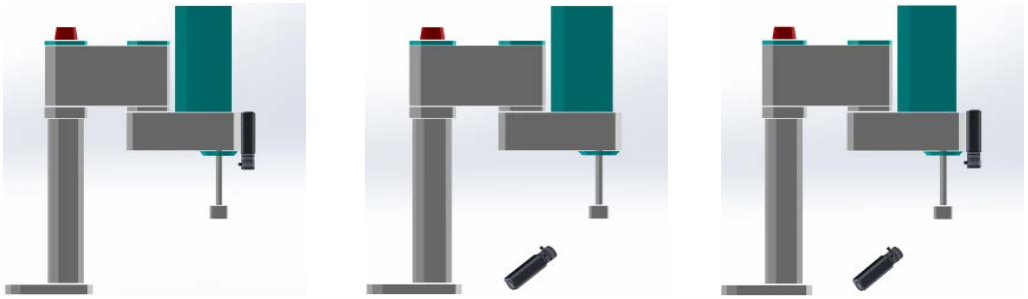


Figura 1. 2: Configuraţiile sistemelor VS: a) sistem servoing „ochi pe mână"; b) sistem servoing „ochi la mână"; c) sistem servoing „mixt".

calibrare a camerei şi modulului de control care se realizează pe baza unor variabile din spaţiul 3D (Vona 2010).

2. În funcţie de numărul de camere video utilizate:
    a. Sisteme servoing monoculare (monocular systems) (Yang, 2007);
    a. Sisteme servoing binoculare (binocular systems) (Cowan, 2002).
3. Din punct de vedere al obiectului vizat:
    a. Sisteme servoing care au ca ţintă vizuală doar obiecte din mediul de lucru se numesc sisteme „obiectiv - buclă deschisă" (endpoint – open – loop systems respectiv EOL);
    b. Sisteme servoing care urmăresc atât obiecte din mediul de lucru cât şi punctul caracteristic al efectorului final al robotului se numesc sisteme „obiectiv - buclă închisă" (endpoint – closed – loop systems respectiv ECL).

Avantajul variantei b („obiectiv - buclă închisă") faţă de prima variantă („obiectiv - buclă deschisă") este dat de precizia ridicată a acestui tip de sistem deoarece nu depinde, ca în primul caz, de corectitudinea şi acurateţea cu care s-a realizat calibrarea robotului şi a camerei. Acest tip de sistem nu depinde practic de erorile pe care le introduce robotul sau camera. Dezavantajul constă în faptul că ambele ţinte, atât obiectul vizat cât şi punctul caracteristic al robotului, trebuie urmărite în mod constant ceea ce duce la creşterea gradului de complexitate al algoritmului de vedere artificială (Fuentes-Pacheco, 2009).

4. Din punct de vedere al schemei de control al robotului:
    a. Sisteme servoing cu control vizual direct (direct visual control systems). Sunt sisteme la care feedback-ul vizual este utilizat în mod direct pentru a realiza controlul cuplelor cinematice ale robotului, lucru ce face ca sistemele de vedere artificială utilizate să fie foarte rapide, deoarece viteza acestora coincide cu viteza de control (Pomares, 2011).
    b. Sisteme servoing cu control vizual indirect (indirect visual control systems). Sunt acele sisteme la care feedback-ul vizual nu este utilizat in mod direct la nivelul acţionării cuplelor cinematice ci este ulterior procesat, fapt ce le face mai lente în comparaţie cu celelalte. Aceste sisteme la rândul lor se împart în două subcategorii:

   i. Sisteme servoing dinamice de tip „priveşte şi acţionează” (dynamic look – and – move systems), sunt acele sisteme la care punctul ţintă al traiectoriei robotului este actualizat în mod continuu pe parcursul mişcării acestuia;
   ii. Sisteme servoing statice de tip „priveşte şi acţionează” (static look – and – move systems) sunt sistemele la care poziţia punctului ţintă este stabilită înainte de a se comanda mişcarea robotului.

5. În funcţie de informaţia vizuală utilizată:
   a. Sisteme servoing bazate pe imagini, denumite şi sisteme de tip 2D (Image – based Visual Servoing systems sau IBVS) (Kim, 2006);
   b. Sisteme servoing bazate pe poziţie, denumite şi sisteme de tip 3D (Position – based Visual Servoing systems sau PBVS) (Cherubini, 2008). Acestea se regăsesc în două categorii:
      i. Sisteme servoing bazate pe poziţie de tip „punct” sau „trăsătură” (point-based PBVS sau features-based PBVS);
      ii. Sisteme servoing bazate pe poziţie de tip „situare” (pose-based PBVS) (Marchand, 2005).
   c. Sisteme servoing hibride denumite şi sisteme de tip 2½D (hybrid systems) (Gans, 2003).

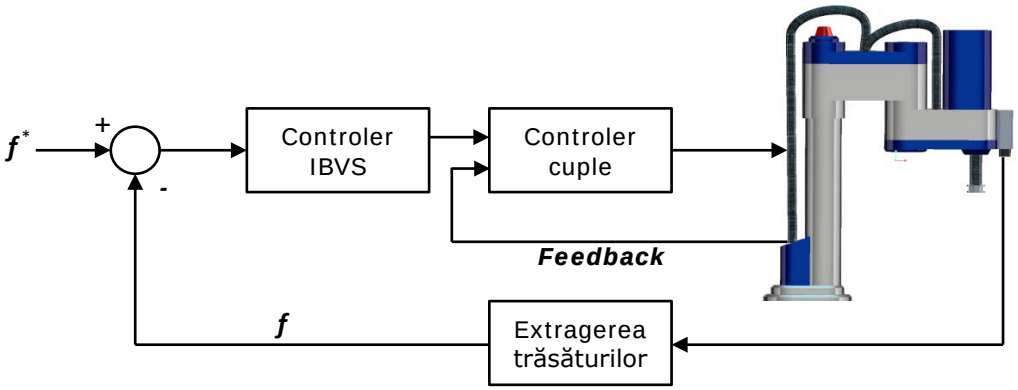


Figura 1. 3: Diagrama unui sistem servoing bazat pe imagine (*f* este vectorul trăsăturilor extrase din imagine, *f** vectorul trăsăturilor de referinţă).

Sistemele servoing bazate pe imagine utilizează informaţiile 2D din imagini pentru a realiza controlul mişcării robotului. Practic acest tip de sisteme utilizează o metodă care pleacă de la nişte parametri de referinţă şi a cărei scop este obţinerea unei erori minime între aceşti parametri şi parametrii extraşi din imagine (Remazeilles, 2004), (Chaumette şi Hutchinson, 2006).

Avantajele acestor sisteme sunt:

- Timpul de procesare al algoritmului este redus;
- Probabilitatea apariţiei unor probleme de vedere computerizată este mai mică.

Dezavantajul principal al sistemelor este:

- Necesitatea existenţei a trei puncte necolineare pentru realizarea legii de control a sistemului.

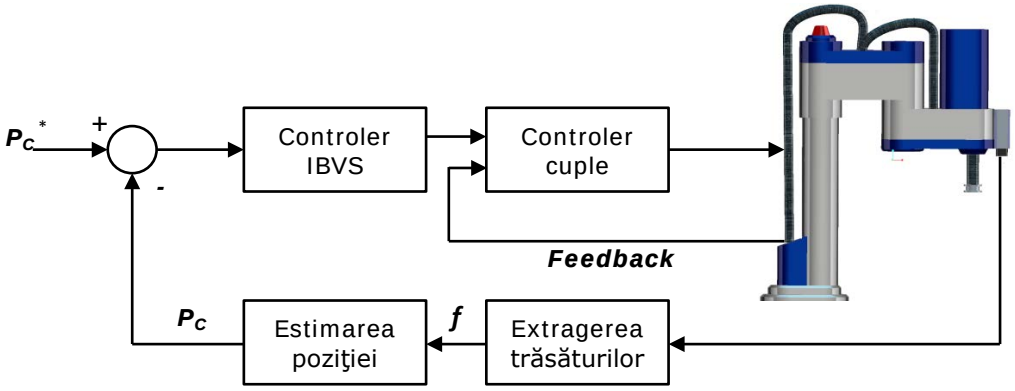


Figura 1. 4: Diagrama unui sistem servoing bazat pe poziţie (f este vectorul trăsăturilor extrase din imagine, f^* vectorul trăsăturilor de referinţă, P situare estimată, P_C^* situarea dorită).

Sistemele servoing bazate pe poziţie se referă la controlul robotului pe baza informaţiilor de natură 3D. Situarea obiectului ţintă se determină utilizând informaţiilor extrase din imagini, care ulterior sunt raportate la informaţiile geometrice ale obiectului şi la informaţiile cunoscute despre cameră, respectiv poziţia şi orientarea acesteia în mediul de lucru (Nomura, 2000).

Avantajele acestor sisteme sunt:

- Precizie ridicată;
- Sisteme robuste.

Dezavantajele sistemelor sunt:

- Necesitatea existenţei unor cunoştinţe despre obiect;
- Calcul laborios (în sistemul cartezian).

Sistemele servoing de tip hibrid sau 2½D sunt de fapt o combinaţie a celor două tipuri de sisteme clasice amintite anterior. Acestea încearcă să păstreze doar avantajele pe care cele două tipuri le presupun, excluzând neajunsurile lor, astfel încât nu utilizează informaţiile de natură 3D despre obiect (ca la PBVS), iar legea de control a robotului nu se realizează pe baza informaţiilor 2D (ca la IBVS) (Rao, 2003)

În urma clasificării sistemelor servoing vizuale, s-a putut observa că indiferent de tipul de sistem ales, nu lipsesc din structura acestuia trei componente esenţiale care sunt prezentate schematic în figura 1.5 (Kragic, 2003)

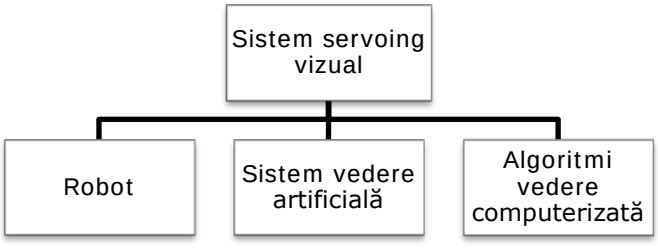


Figura 1. 5: Componenţa unui sistem servoing

1.3. Structura şi conţinutul tezei de doctorat

Teza de doctorat intitulată „Contribuţii la utilizarea prelucrării automate a imaginii la roboţi de manipulare" se întinde pe 187 de pagini şi este structurată pe 9 capitole. Teza mai cuprinde o bibliografie de 187 de titluri în marea lor majoritate de dată recentă, 87 de figuri, 18 tabele, precum şi 11 anexe.

Capitolul 1, „Introducere", tratează aspecte generale cu privire la utilizarea vederii artificiale în domeniul roboticii, o clasificare a sistemelor de servoing vizual, avantajele acestora şi prezintă succint structura tezei de doctorat.

Capitolul 2, „Stadiul actual", prezintă o sinteză bibliografică cu privire la domeniul de activitate în care se încadrează tema de cercetare, respectiv aspecte generale cu privire la roboţii industriali, tipuri de roboţi utilizaţi frecvent în aplicaţii de servoing vizual şi avantajele acestora. De asemenea se prezintă noţiuni cu privire la sistemele de vedere artificială, componenţa acestora, metode de calibrare a camerelor video, algoritmi de vedere coputerizată utilizaţi frecvent în aplicaţii de vedere robotizată, metode de prelucrare a imaginilor ce stau la baza acestor algoritmi şi aplicaţii de vedere robotizată existente la ora actuală în industrie.

Capitolul 3, „Obiectivele şi planul de cercetare", schiţează principalele direcţii de cercetare ale prezentei teze, motivaţia abordării temei, obiectivele şi planul de cercetare propus.

Capitolul 4, „Aplicaţie de inspectare a suprafeţelor", prezintă o aplicaţie de vedere robotizată concepută pentru realizarea unei operaţii de inspectare ce presupune detectarea, identificarea şi determinarea tipului de fisură sau crăpătură de la nivelul suprafeţei analizate.

Capitolul 5, „Proiectarea de aplicaţii în LabView pentru procesarea imaginilor şi controlul roboţilor" descrie modalitatea de elaborare a două aplicaţii, una de servoing vizual şi cealaltă de vedere computerizată, utilizând mediul de lucru LabView şi camere video industriale de tip „smart camera".

Capitolul 6, „Aplicaţie de detectare şi recunoaştere a unor obiecte în vederea manipulării robotizate" tratează detectarea şi recunoaşterea de obiecte pe bază de vedere artificială şi prezintă utilizarea unei algoritm, dezvoltat de autor, în cadrul unei aplicaţii de servoing vizual al cărui scop îl reprezintă manipularea unor obiecte colorate pe baza unei caracteristici geometrice.

Capitolul 7, „Conceperea şi dezvoltarea unor aplicaţii industriale de tip „pick and place" în Matlab", prezintă implementarea pe o structură robotizată, a trei aplicaţii industriale concepute şi dezvoltate pentru identificarea, manipularea şi sortarea unor piese industriale.

Capitolul 8, „Aplicaţie de recunoaştere şi prehensare a obiectelor" tratează implementarea într-o aplicaţie de vedere robotizată, a unei noi metode de determinare a punctelor de prehensare optime, în cazul unor obiecte necunoscute, pentru realizarea unei manipulări stabile, de către roboţii industriali.

Capitolul 9, „Concluzii, contribuţii originale şi direcţii viitoare de cercetare" prezintă concluziile finale, contribuţiile originale şi perspectivele privind dezvoltarea cercetărilor în domeniu.

2. STADIUL ACTUAL

În cadrul acestui capitol, se prezintă, o sinteza bibliografică cu privire la domeniul de activitate în care se încadrează tema de cercetare abordată, respectiv, aspecte generale cu privire la roboţii industriali şi sisteme de vedere artificială. Se tratează de asemenea metode de calibrare a camerelor video, algoritmi de vedere computerizată utilizaţi frecvent în aplicaţiile de vedere robotizată, metode de prelucrare a imaginilor şi aplicaţii de vedere robotizată existente la ora actuală în industrie.

2.1. Roboţi industriali

În urma studierii literaturii de specialitate, se poate afirma că mai multe instituţii din întreaga lume, precum JIRA (Japan Industrial Robot Association) sau RIA (Robot Institute of America), au încercat să dea o definiţie cât mai clară şi acoperitoare cu privire la semnificaţia termenului de robot industrial (Kovacs, 2000), (VISION, 2012).

Varianta cea mai acceptata şi utilizată, este cea dată de Institutului de Standardizare Internaţional (ISO), care defineşte robotul industrial ca „un manipulator controlat automat, reprogramabil, multifuncţional, programabil pe trei sau mai multe axe, care poate fi fix sau mobil şi este utilizat în aplicaţii de automatizări industriale" (Austin, 2005), (Mathia, 2010).

Ştiinţa care se ocupă cu studiul, construcţia şi utilizarea roboţilor poartă numele de „robotică". Această ştiinţă reprezintă un vast domeniu interdisciplinar, ce utilizează noţiuni din aria ingineriei mecanice, electrice şi ştinţa calculatoarelor.

Roboţii industriali au evoluat foarte mult (vezi anexa 1) de la primul robot cu acţionare hidraulică, proiectat în anul 1954 de George Devol, şi comercializat ulerior în 1961, până la roboţii moderni, din ziua de astăzi (figura 2.1), flexibili, prevăzuţi cu sistem de vedere artificială şi un controler inteligent, cum este robotul industrial Frida de la firma suedeză ABB (Kovacs, 2000), (Mathia, 2010), .

Demn de menţionat, este şi faptul că, în România apare primul robot industrial „Rip 6.3" de tip ASEA, în 1980 construit la întreprinderea Automatica Bucureşti sub coordonarea I.C.P.T.C.M. Bucureşti în cadrul unui program naţional (Iordăchiţă 1997).

În anul 1982, la Timişoara, în cadrul institutului Politehnic „Traian Vuia" apare primul robot de concepţie integrală românească „REMT-1" realizat de către colectivul de robotică condus de regretatul profesor Francisc Kovacs. Acest robot a fost folosit la Electromotor Timişoara în cadrul unei celule de prelucrare prin aşchiere a unei familii de arbori (Richard, 2005), (Kruachottikul, 2012).

În 1987 Institutul Politehnic „Traian Vuia" Timişoara şi IMMUM Baia Mare realizează un manipulator sincron – MS 500 expus la TIB´87 şi distins cu Medalia de aur iar anul următor realizează un robot portal „ROPOS 50" Expus la TIB´88 şi distins cu tot Medalia de aur (Iordăchiţă, 1997).

(Romeo, 1996) prezintă robotul ca „o componentă evoluată de automatizare, ce combină electronica de tip calculator, cu sisteme avansate de acţionare mecanică, pentru a realiza, în final un echipament independent de mare

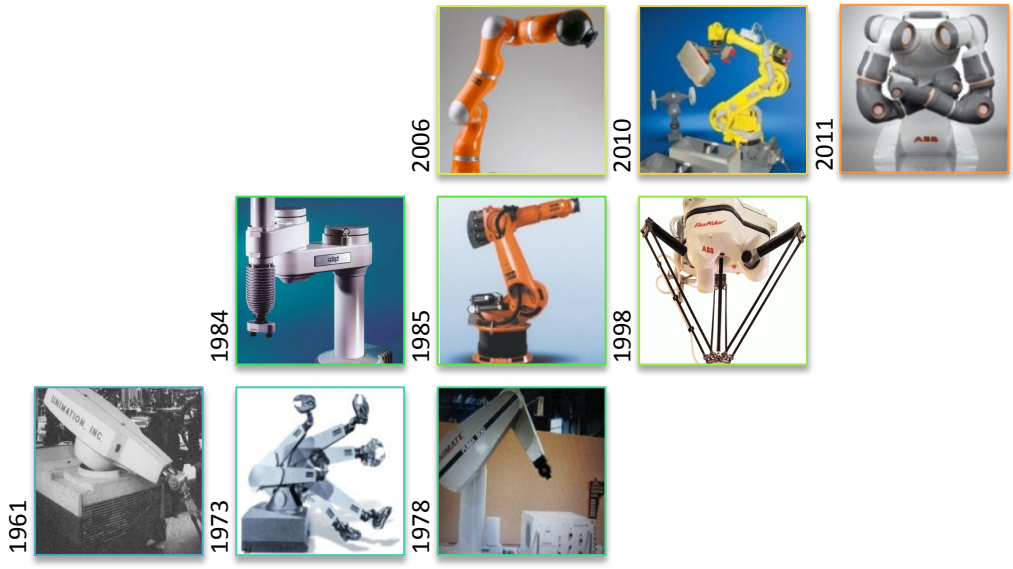


Figura 2. 1: Evoluţia roboţilor industriali.

flexibilitate". Pe baza acestei definiţii, se poate afirma că, robotul este un produs „mecatronic" (Romeo, 1996), (VISION 2012).

În prezent, acest produs mecatronic este în continuă dezvoltare, fapt ce duce la creşterea numărului şi varietăţii acestora. O valoare aproximativă a roboţilor industriali funcţionali la nivel mondial în diverse domenii de activitate şi achiziţionaţi anual este prezentată schematic în diagrama din figura 2.2.

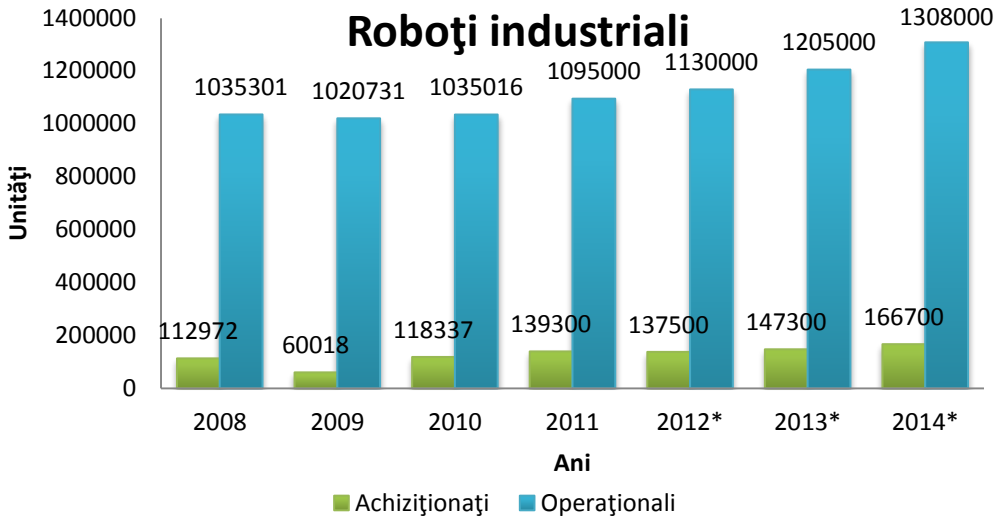


Figura 2. 2: Diagramă cu roboţii industriali achiziţionaţi şi operaţionali la nivel mondial (sursa: IFR Statistical Department; * estimare).

Structura generală a unui robot industrial cuprinde (figura 2.3):

Sistemul de comandă al robotului este reprezentat de ansamblul echipamentelor şi dispozitivelor de calcul (procesoare, microprocesoare, s.a.) care preia informaţie de la sistemul senzorial şi de la operatorul uman şi transmite semnale de comandă sistemului de acţionare.

Sistemul de acţionare constă în acele componente ale robotului industrial (motoare, transmisii mecanice s.a.) care transformă un anumit tip de energie (de natura pneumatică, hidraulică sau electrică) în energie mecanică.

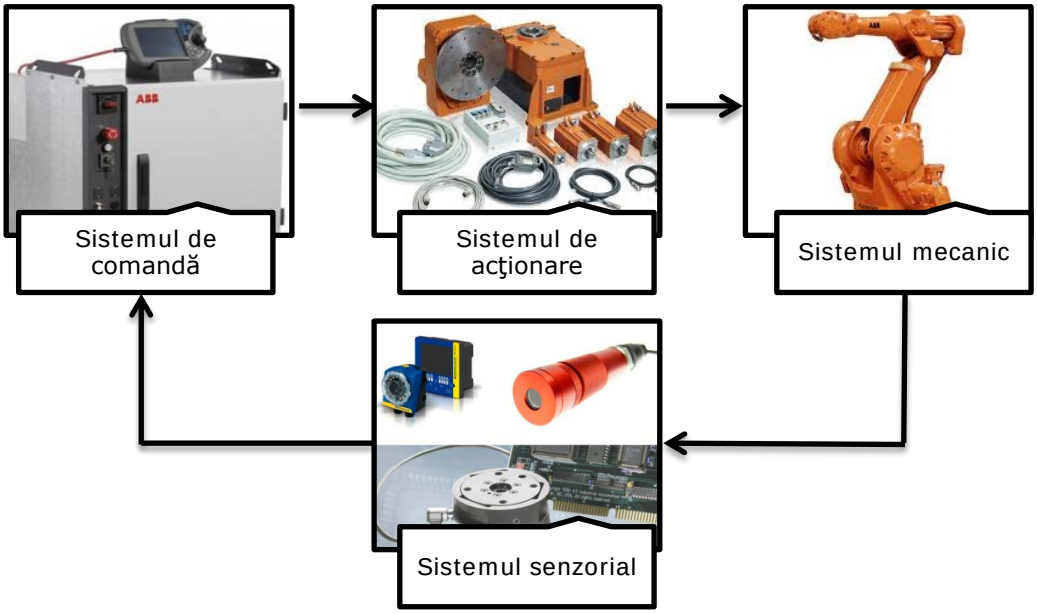


Figura 2. 3: Structura generală a unui robot industrial.

Sistemul mecanic al robotului industrial se identifică cu robotul propriu-zis respectiv cu structura mecanică a acestuia şi este alcătuit din mai multe elemente legate între ele prin cuple cinematice care asigură mişcarea efectorului final al robotului.

Sistemul senzorial al robotului industrial este reprezentat de totalitatea senzorilor (de tip vizual, tactil, sonor, ş.a.) şi traductoarelor (de deplasare, forţă, presiune, ş.a.) ce prelevează informaţii cu privire la mediul exterior al robotului sau la starea internă de funcţionare a robotului.

Sistemul mecanic al robotului este compus la rândul său din următoarele subsisteme:

- Dispozitivul de ghidare (partea articulată a robotului) care asigura realizarea mişcărilor efectorului final;
  - Mecanismul generator de traiectorie (MGT) numit şi braţ este lanţul cinematic deschis care asigură poziţia terminalului;
  - Mecanismul de orientare (MO) numit şi încheietură este lanţul cinematic deschis care asigură orientarea terminalului.
- Dispozitivul de prehensiune numit şi organ terminal este un dispozitiv de cuplare ce permite robotului să apuce piesa.

Această mulţime de roboţi industriali este clasificată, în literatra de specialitate, după mai multe criterii (Iordăchiţă 1997), (Telea 2002), (Robotics 2005), (Zisopol, 2006), (Rădulescu, 2008).

O clasificare a roboţilor industriali în funcţie de principalele domenii de activitate este prezentată în figura 2.4.

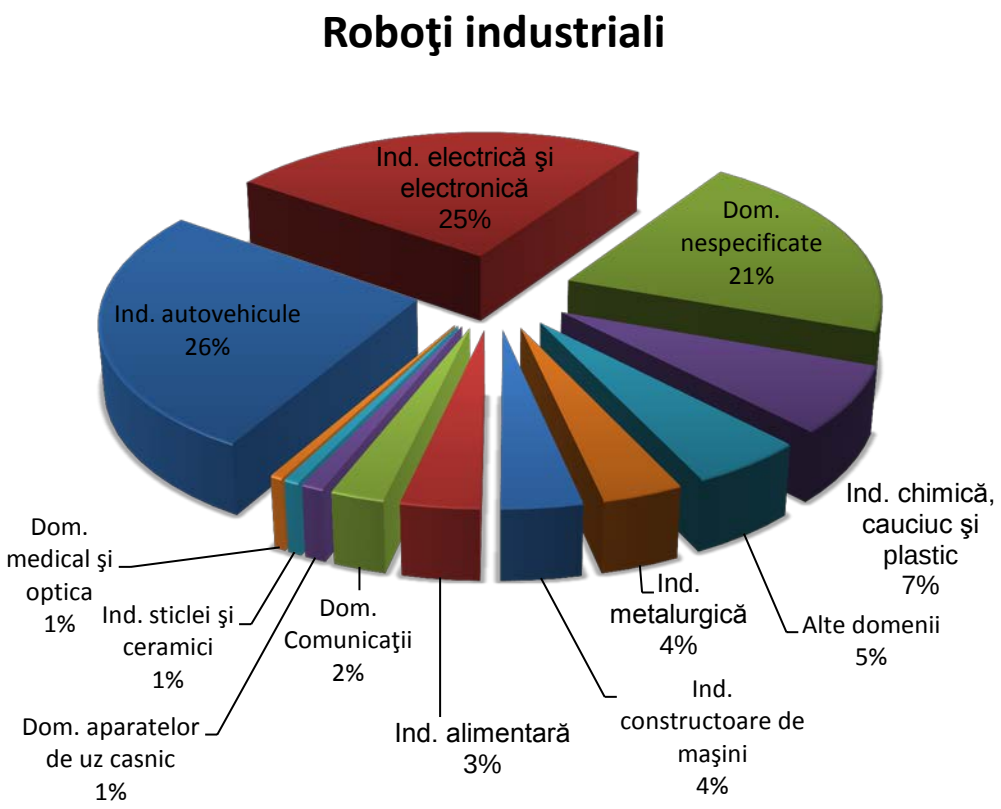


Figura 2. 4: Diagramă cu roboţii industriali în principalele domenii de activitate

O clasificarea a roboţilor industriali în funcţie de tipul procesului tehnologic respectiv de tipul aplicaţiei efectuate conform cu (Romeo 1996), (Zisopol 2006), (Rădulescu, 2008) este constituită din:

- Roboţi pentru manipulare sau deservire a maşinilor unelte (38%):
  - Operaţii pentru turnare metalică;
  - Operaţii pentru deformare plastică;
  - Operaţii de ştanţare, presare şi forjare;
  - Operaţii de alimentarea automată cu piese, scule sau dispozitive a maşinilor unelte;
  - Operaţii de control automat al dimensiunilor şi formei piesei de lucru;
  - Operaţii de împachetare şi ambalare;
  - Operaţii de încărcare şi descărcare a conveioarelor;
  - Operaţii de transport şi depozitare/înmagazinare;

- Roboţi utilizaţi în procese tehnologice de acoperiri superficiale (vopsire, decapare 4%);
- Roboţi pentru procese tehnologice de sudare şi lipire (29%);
- Roboţi pentru procese tehnologice de prelucrări mecanice (aşchiere) (2%);
- Roboţi pentru procese tehnologice de asamblare şi dezasamblare (10%);

Roboţii industriali se diferenţiază pe baza unor caracteristici şi coeficienţi de performanţă. Principalele caracteristici ale acestor roboţi sunt (Zisopol, 2006):

- Dimensiunile gabaritice ale robotului industrial;
- Gradul de mobilitate;
- Capacitatea de ridicare;
- Precizia de poziţionare;
- Viteza de deplasare;
- Numărul gradelor de libertate;
- Spaţiul de lucru;
- Valoarea deplasării realizate;
- Repetabilitatea realizării unei poziţii;
- Capacitatea sistemului de comandă şi control.

Coeficienţii de performanţă sunt daţi de trei indici ce se calculează utilizând volumul spaţiului de lucru (V_{SP}), masa robotului (M_R), masa obiectului manipulat (M_{OB}) şi precizia de poziţionare (P_{POZ}):

- Indicele de eficienţă şi supleţe: $k1 = \frac{V_{SP}}{M_R}$
- Indicele capacităţii de manipulare: $k2 = \frac{M_{OB}}{M_R}$
- Indicele global: $k3 = \frac{V_{SP} * M_{OB}}{M_R * P_{POZ}}$

Utilizarea unui robot în industrie conduce la o serie de avantaje dar şi la dezavantaje:

- Creşterea calităţii produselor: se datorează faptului că robotul industrial poate realiza operaţii repetitive cu precizie şi acurateţe ridicată;
- Eficienţă: este dată de faptul că un robot poate realiza o operaţie tehnologică cu o viteză foarte mare fără oprire, fapt ce conduce la creşterea producţiei;
- Siguranţă: Utilizarea robotului în industrie conduce la un grad ridicat de siguranţă al omului datorită faptului că acestuia îi revine doar rolul de supraveghetor, fiind scutit de realizarea unor activităţi grele din punct de vedere fizic sau lucru în medii toxice ce prezintă un pericol pentru sănătate.
- Economie: se referă la faptul că un robot industrial este mai rapid, ceea ce implică un timp de producţie scăzut şi la faptul că este mai precis, ceea ce face ca numărul produselor rebut să fie scăzut.
- Cheltuieli ridicate: costurile pentru achiziţie, mentenanţă şi instruirea unui personal sunt ridicate;
- Complexitatea operaţiilor: gradul de adaptabilitate a robotului la diverse sarcini industriale este limitat datorită faptului că acesta nu poate realiza decât acele operaţii pentru care a fost proiectat.
- Rezolvarea problemelor: se referă la incapacitatea robotului de a găsi soluţii în anumite situaţii ceea ce îl face dependent de gândirea, creativitatea şi inovaţia omului.

2.2. Sisteme de vedere artificială

Vederea artificială (V.A.) se poate defini ca un proces ce permite achiziţionarea imaginilor, din mediul înconjurător, extragerea informaţiei utile din imagini şi procesare ei (Gonzalez, 2002). Acest proces în general presupune parcurgerea următoarele etape:

- *Achiziția imaginii;*
- *Prelucrarea primară* are drept scop îmbunătăţirea calităţii imaginii, amplificarea contrastului şi reducerea sau eliminarea artefactelor şi zgomotului;
- *Segmentarea* permite realizarea operaţiei de descompunere a imaginii în diferite obiecte de interes;
- *Descrierea* este procesul prin care informaţia necesară separării obiectelor este redată prin intermediul diferitelor caracteristici şi trăsături.
- *Recunoaşterea* este etapa prin care se realizează identificarea şi clasificarea obiectelor pe baza trăsăturilor;
- *Interpretarea* este procesul în care obiectelor recunoscute le sunt atribuite diferite semnificaţii.

Etapele descrise anterior, în funcţie de gradul de complexitate, se grupează în trei categorii ce sunt ilustrate în figura 2.5.

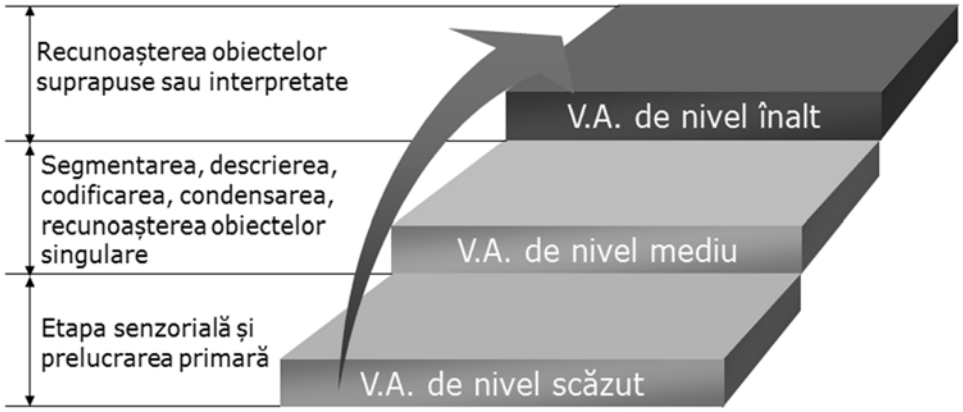


Figura 2. 5: Niveluri ale vederii artificiale.

Implementarea acestui proces, a presupus realizarea unor sisteme de vedere artificială. Crearea şi dezvoltarea lor s-a efectuat pe baza unei analogii cu sistemele biologice, în general, respectiv cu sistemul uman, în particular. Structura unui sistem de vedere artificial este compusă în principiu din:

- Una sau mai multe camere video, cu sistem optic adecvat și cu elemente constitutive precum tubul vidat sau senzori de imagine de tip CMOS sau CCD cu electronica lor aferentă;
- Interfața pentru digitizarea imaginilor;
- Unitate de procesare;
- Intrări şi ieşiri hardware (Input / Output) sau rețele de comunicare pentru a raporta rezultatele;

- Lentile;
- Sursă de lumină (led-uri, lampă fluorescentă sau halogenă);
- Un program capabil să proceseze imaginile și să detecteze caracteristicile esențiale;

Pe lângă acestea un sistem de vedere artificială mai poate să conțină:

- Senzor de sincronizare pentru detectarea părților componente, declanșarea înregistrării imaginilor și procesării;
- Dispozitive pentru a sorta și elimina articolele cu defecte.

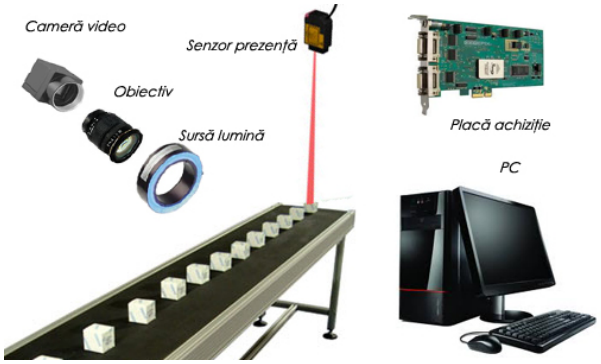


Figura 2. 6: Sistem de vedere artificial.

O clasificare a sistemelor de vedere artificiale ce se utilizează în industrie poate fi:

- Sisteme de vedere artificială cu aplicabilitate generală, bazate pe un calculator personal (un PC);
- Sisteme de vedere artificială cu aplicabilitate specifică, compacte;
- Sisteme de vedere artificială bazate pe reţele de camere video.

Aceste sisteme de vedere artificială variază între ele din punct de vedere constructiv şi funcţional. Cele mai importante aspecte de care trebuie ţinut cont la proiectarea, construcţia sau achiziţionarea unui astfel de sistem sunt sintetizate în tabelul 2.1:

Cameră video	Unitatea de procesare	Sursa de lumină
Tipul camerei: – monocromă; – color; Tipul semnalului: – analogic – digital; Tipul obiectivului: – superangulare; – teleobiective; – focală normală; – "fish-eye"; – "tilt&shilt" Tipul senzorului vizual.	Puterea de calcul. Capacitatea de stocare. Memoria de lucru. Tipul softului de vedere computerizata. Tipul plăcii de captură: – Porturi de intrare\ieşire; – Rata de eşantionare; – Precizia; – Zgomot;	Tipuri surse de iluminat; Modalităţi de iluminare: – difuză; – direcţionată; – laterală; – polarizată; – structurată.

Tabelul 2. 1: Aspecte generale ale sistemelor de vedere artificială.

În unele cazuri toate dispozitivele enumerate mai sus sunt combinate într-unul singur numit cameră inteligentă.

Parametri fundamentali ai unui astfel de sistem sunt (Edmund_Optics 2012), (Gruescu, 2012).

- Câmpul vizual (field of view - FOV) se defineşte ca fiind zona vizibilă a mediului sau obiectului ce este prelevată de senzorul vizual;
- Profunzimea câmpului (depth of field - DOF), reprezintă adâncimea maximă a obiectului ce poate fi focusată;
- Distanţa de lucru (working distance - WD) este distanţa de la lentilă până la obiectul sau suprafaţa inspectată;
- Rezoluţia (resolution) reprezintă distanţa minimă dintre două entităţi (puncte) ale obiectului inspectat ce se pot distinge separat;
- Deschiderea sau apertura obiectivului, caracteristică ce determină cantitatea de lumină captată de senzor;
- Mărimea senzorului (sensor size) dimensiunea zonei active a senzorului măsurată orizontal;

Alţi factori pot fi:

- Mărirea primară (primary magnification - PMAG) se defineşte ca raportul dintre aria senzorului şi câmpul vizual;
- Timpul de expunere, reprezintă o măsură a vitezei obturatorului (*shutter speed*) ce se măsoară în secunde sau fracţiuni de secundă;

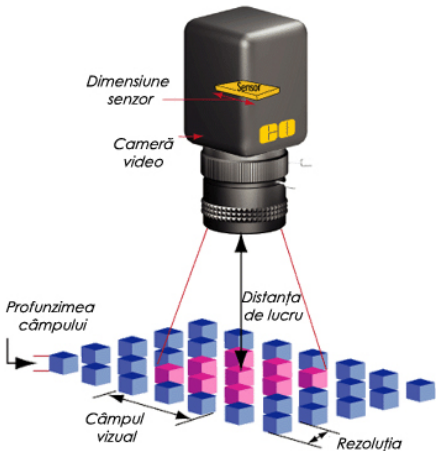


Figura 2. 7: Parametri fundamentali ai unui sistem de vedere (Edmund_Optics, 2012).

Prin urmare, în conformitate cu cele menţionate anterior, se poate deduce că cea mai importantă componentă a oricărui sistem de vedere artificial este camera video. Aceasta este prevăzută cu un senzor vizual ce converteşte lumina în semnal electric pe baza efectului fotoelectric.

„Senzorul vizual este un dispozitiv optoelectronic sensibil la componenta radiometrică şi spectrală a radiaţiei electromagnetice, în domeniul vizibil. El realizează prelevări statice sau dinamice ale unei imagini colorate sau în nuanţe de gri. Funcţie de tehnologia de stocare şi redare, imaginea poate fi digitală sau analogică. În marea majoritate a aplicaţiilor din prezent, imaginile au un caracter digital" (Gruescu, 2012).

În funcţie de tehnologia utilizată la realizarea acestor senzori vizuali, există mai multe tipuri. Cei mai răspândiţi la ora actuală sunt cei sub formă de matrice de celule fotosensibile, ce poate fi o matrice de tipul CCD (charged couple device) sau o matrice CMOS (complementary metal oxide semiconductor). Fiecare element al matricei, corespunzător unui pixel al imaginii, preia o componentă informaţională fotometrică (prezenţa la diferite niveluri a energiei luminoase sau absenţa acesteia) şi la senzorii de culoare, suplimentar, şi componentele unui sistem de coordonate de culoare (Gruescu, 2012).

Senzorul electronic CCD a fost inventat de George Smith şi Willard Boyle în 1969, invenţie pentru care au primit *Premiul Nobel* în 2006. CCD este primul dispozitiv de acest tip apărut pe piaţă şi este de fapt un circuit integrat pe bază de siliciu format dintr-o matrice densă de celule capacitive, care transformă energia luminoasă în semnal electric.

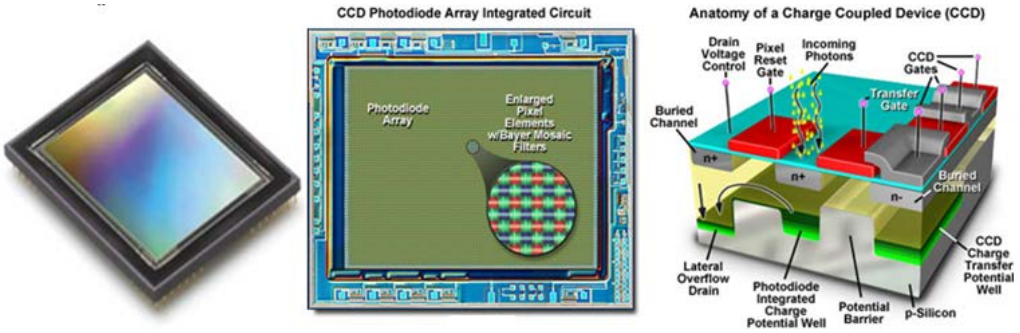


Figura 2. 8: Senzorul CCD (adaptat după (Dalsa, 2012), (Spring, 2012), (Downhill, 2012)).

Caracteristicile principale ale senzorului CCD ce trebuie avute în vedere pentru realizarea unui sistem de vedere artificială sunt:

- Numărul pixelilor (este egal cu numărul fotocelulelor senzorului);
- Dimensiunile pixelilor (sunt de ordinul micrometrilor 2.2 …11µm);
- Aria senzorului (este dată de numărul pixelilor de pe orizontală şi numărul pixelilor de pe verticală);

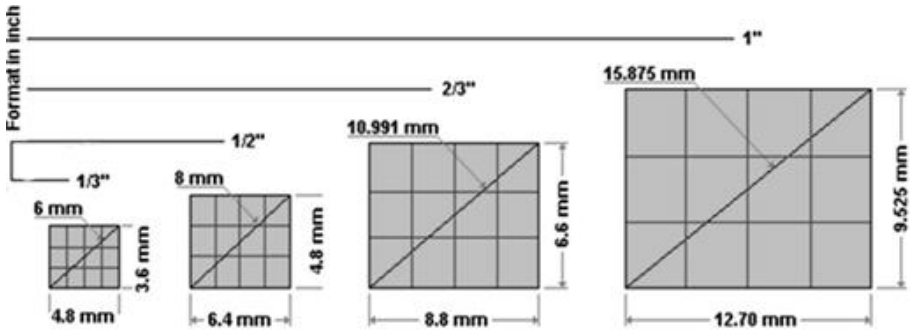


Figura 2. 9: Variante arie senzor (adaptat după (Secuv, 2012))

- Formatul optic (este diagonala maximă a imaginii proiectată în planul focal);

Format	Pixeli
QVGA	320x240
CGA	600x200
EGA	600x350
VGA	640x480
SVGA	800x600
XGA	1024x768
SXGA	1280x1024
UXGA	1600x1200
WUXGA	1920x1200
QZGA	2084x1536

Tabelul 2. 2: Tipuri de format optic.

- Rezoluţia (este distanţa minimă dintre doi pixeli, fiind practic dublul lăţimii unui pixel);
- Contrastul;
- Raportul semnal/zgomot;
- Responsivitatea (măsura capacităţii de producere a unui semnal electric);
- Adâncimea pixelului (gama dinamică sau scala de griuri ce reprezintă numărul de nuanţe de gri ce pot fi redate în planul imaginii);
- Semnalul de ieşire (analogic sau digital).

Un alt tip de senzor vizual este senzorul CMOS. Senzorul CMOS (complementary metal oxide semiconductor) este mai nou decât senzorul CCD. În primă instanţă a apărut doar ca o variantă mai ieftină a senzorului CCD, ce deţinea un nivel calitativ inferior al imaginii, dar care se putea realiza utilizând o tehnologie de fabricaţie mai accesibilă producătorilor. Ulterior, odată cu dezvoltarea şi maturizarea tehnologiei, acesta a devenit un concurent serios predecesorului lui. În comparaţie cu senzorul CCD, senzorul CMOS are pentru fiecare fotocelulă (fotodiodă) câțiva tranzistori miniaturizați, care au rolul de a amplifica și converti semnalul analogic în semnal digital și de a transfera informația prin conexiuni standard. Avantajul pe care un astfel de senzor îl deţine, în comparaţie cu un senzor CCD, este dat de consumul de energie scăzut (aproximativ de o suta de ori mai mic). Dezavantajul lor este sensibilitatea luminoasă scăzută, datorită bombardării simultane de către fotoni a fotocelulelor şi tranzistorilor.

Camerele de luat vederi (camerele video) din ziua de astăzi, fie analogice sau digitale, clasice sau inteligente, în funcție de producător şi de performanţele scontate, încorporează unul din cei doi senzori amintiţi anterior. De asemenea o cameră de luat vederi inteligentă pote încorpora toate componentele pe care le deţine un sistem de vedere artificial.

Camerele inteligente (smart cameras) au apărut relativ recent (la începutul anilor nouăzeci) în industrie, datorită faptului că erau limitate din punct de vedere al capacităţilor de procesare a imaginilor, fiind utilizate pentru aplicaţii simple, precum citirea codurilor de bare (ThomasNet, 2012). Acest lucru s-a schimbat rapid odată cu dezvoltarea proceselor de fabricaţie a circuitelor integrate şi creşterea puterii de calcul a microcontrolerelor şi procesoarelor.

O posibilă definiţie a unei camere inteligente ar fi camere ce înglobează un sistem de vedere artificial clasic fiind capabile să extragă informaţii din imaginile prelevate, în funcţie de aplicaţie, astfel încât să ia o anumita decizie ce poate fi folosită la conducerea unui sistem automat.

Camerele inteligente, ce sunt produse în prezent şi utilizate în special în industrie, se pot clasifica după gradul de integrare a elementelor componente conform figurii 2. 10.

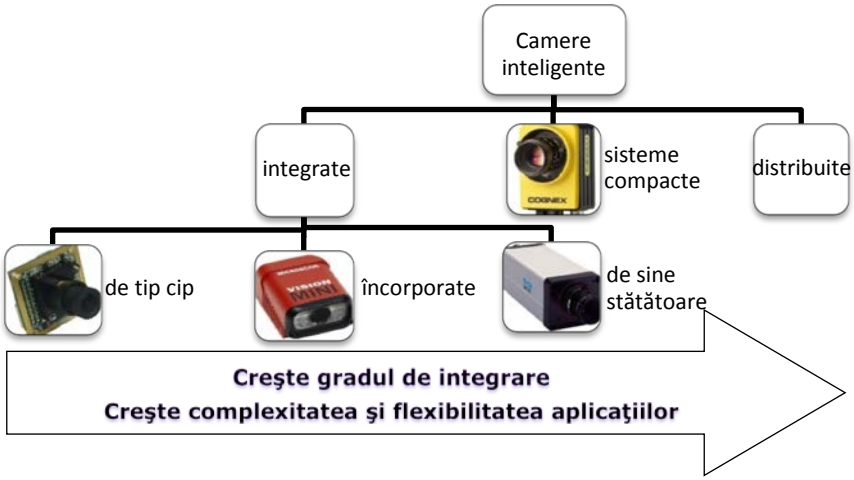


Figura 2. 10: Clasificarea camerelor inteligente (adaptat după(Yu Shi, 2010))

În tabelul 2.3 sunt prezentate avantajele şi dezavantajele sistemelor de vedere artificială bazate pe PC şi cele de tip cameră inteligentă (Yu Shi, 2010).

	Avantaje	**Dezavantaje**
Camere inteligente	Dimensiuni mici; Cost de achiziţie relativ mic; Compacte; Arhitectură simplă; Dispozitive de conexiune şi comunicare integrate; Cost de întreţinere mic; Soft simplu şi accesibil tuturor; Generează cantitate mică de căldură; Robuste în exploatare.	Capacitate de procesare limitată; Performanţe reduse; Aplicabilitate restrânsă; Schimbare de piese dificilă.
Sisteme bazate pe PC	Putere de procesare mare; Îmbunătăţirea performanţelor; Aplicabilitate extinsă; Schimbare de piese uşoară; Complexitatea mare a aplicaţiilor.	Dimensiuni relativ mari; Cost de achiziţie mai ridicat; Necesită componente multiple; Număr ridicat de tipuri de interfeţe pentru conectare; Softuri complexe şi dificile; Generează o cantitate mai mare de căldură; Robusteţe mică în exploatare.

Tabelul 2. 3: Avantajele şi dezavantajele sistemelor de vedere artificială.

2.3. Algoritmi de vedere computerizată

Vederea computerizată reprezintă domeniul de cercetare care se ocupă cu dezvoltarea metodelor ce permit achiziţia, procesarea şi analiza imaginilor prelevate din mediul ambiant, în vederea extragerii, caracterizării şi interpretării informaţiilor, a căror finalitate constă în luarea unei decizii şi realizarea unei acţiuni. Scopul acestor metode, este de înzestrare a sistemelor de inginerie, cu capacitatea de a interacţiona la nivel vizual cu mediul înconjurător, prin utilizarea sistemelor de vedere artificială (Bradski, 2008).

Algoritmii de vedere computerizată, ce se regăsesc în prezent, implementaţi în cadrul diverselor sisteme robotizate, utilizaţi pentru realizarea unor aplicaţii de vedere robotizată, presupun în general operaţii precum:

- Detectarea obiectelor („object detection”);
- Identificarea obiectelor („identifying objects”);
- Recunoaşterea obiectelor („object recognition”);
- Clasificarea obiectelor („classifying objects”);
- Localizarea obiectelor („locating objects”);
- Urmărirea obiectelor („tracking objects”);
- Inspectarea obiectelor („inspecting objects”).

Operaţia de detectare a unui obiect ţintă, într-o imagine, este o etapă importantă în orice aplicaţie de vedere robotizată. Gradul de dificultate al acesteia, variază în funcţie de complexitatea obiectului vizat şi complexitatea imaginii.
Algoritmii de detectarea a obiectelor, în imagine, se bazează în general, pe detectarea unor trăsături locale sau globale. Trăsăturile locale sunt extrase şi evaluate, în jurul unei locații (pixel) din imagine şi pot consta în nivel de gri, culoare, muchie, textură. Trăsăturile globale pot fi determinate utilizând o histogramă normalizată, ce estimează distribuţia de probabilitate a nivelurilor de gri în imagine, respectiv permite stabilirea minimului şi maximului histogramei, entropia histogramei, media de gri şi dispersia.

Metodele cele mai frecvent utilizate, la extragerea trăsăturilor, necesare diverselor operaţii de vedere computerizată, sunt:

- Metoda SIFT (Scale Invariant Feature Transform –transformata trăsături invariante la scalare) permite detectarea trăsăturilor locale ce nu se modifică în imagine în urma aplicării unor operaţii geometrice precum scalare, rotire sau translatare (Lowe, 2004).
- Metoda SURF (Speeded Up Robust Feature) permite, asemănător metodei SIFT, detectarea unor trăsături locale, invariante la operații geometrice, într-un mod mai robust şi mai rapid (Bay, 2006), (Bay, 2008).
- Metoda ASIFT (Affine Scale Invariant Feature Transform) aduce îmbunătățiri metodei SIFT datorită faptului că estimează alţi doi parametri necesari detectării trăsăturilor locale prin utilizarea transformatelor afine (Guoshen, 2009).

În literatura de specialitate, se regăsesc o mulţime de algoritmi de detectare, ce se pot împărţi în trei categorii, din punct de vedere al metodei de proiectare (Wang, 2007):

- Metode ascendente (Bottom_Up), la care operația de detectare se realizează utilizând trăsături de tip muchie sau segment. Se stabilesc caracteristicile celor mai reprezentativi pixeli sau zone din imagine, şi se potrivesc cu cele învăţate pentru obiect. Pe baza lor, se construiesc anumite ipoteze şi condiţii ce permit evaluarea ulterioară a obiectelor.

- Metode descendente (Top_Down), la care operaţia de detectare se realizează cu ajutorul unui model, anterior învăţat, ce corespunde obiectului ţintă. Se împarte imaginea în regiuni şi se compară fiecare regiune cu şablonul obiectului.
- Metode mixte.

În funcţie de aplicaţie, operaţia de detectare a obiectelor se realizează utilizând diverse abordări. Dintre aceste sunt amintite următoarele:

- Detectarea pe bază de nuanţă sau de culoare, respectiv pe bază de histogramă, este o tehnică foarte des întâlnită în cadrul algoritmilor de vedere computerizată. Se utilizează cu precădere la imagini ce prezintă o calitate slabă şi o rezoluţie scăzută, unde trăsătura fundamentală a obiectului vizat o reprezintă culoarea. Detectarea obiectului ţintă, se realizează, prin stabilirea unei condiţii, pe baza unor valori de prag, corespunzătoare obiectului, ce sunt impuse histogramelor componentelor spaţiului de culoare utilizat. Din imagine, sunt extraşi pixelii, ce îndeplinesc condiţia de prag.(Bradski, 2008).
- Detectarea cu ajutorul unei ferestre glisante (Sliding Window) care se află în conformitate cu un model sau o trăsătură, ce parcurge întreaga imagine analizată de mai multe ori (aplicând un factor de scalare), astfel încât să clasifice fiecare zonă ca obiect ţintă sau fundal. Această metodă este folosită, în special, la detectarea obiectelor compacte (Henry, 1995), (Papageorgiou, 2000), (Schneiderman, 2000), (Viola, 2004).
- Detectarea cu ajutorul unui descriptor global corespunzător obiectului vizat, învăţat într-o etapă premergătoare. În acest caz, imaginea analizată, se împarte într-un număr de regiuni, şi pentru fiecare regiune se stabileşte un descriptor global utilizând metoda GIST (Global Invariant Scale Transform). Obiectul se determină prin compararea descriptorului învățat cu descriptorii regiunilor (Torralba, 2008).

Operaţia de recunoaştere, este un proces ce presupune identificarea unui anumit obiect într-o imagine, pentru ca acesta să fi ulterior clasificat. Procesul de recunoaștere, se bazează, în general, pe diverşi algoritmi de vedere computerizată, care în principiu presupun anumite cunoştinţe despre aspectul obiectelor ce urmează a fi detectate în imaginile analizate. La ora actuală, există o mare varietate de algoritmi de recunoaştere, fapt ce se datorează, în principiu, cerinţelor şi constrângerilor specifice aplicațiilor în care sunt utilizaţi. O scurtă clasificare a celor mai utilizați algoritmi de acest tip, poate fi următoarea:

- Algoritmi ce se bazează pe un model al obiectului definit cu ajutorul unor descriptori globali:
  - Algoritmi care au la bază metoda suprapunerii de şablon (Template Matching), respectiv algoritmi care presupun realizarea unei corelaţii între un şablon al obiectului ţintă şi regiuni din imaginea analizată. Corelaţia dintre cele două reprezintă o măsura a gradului de potrivire. Aceşti algoritmi pot fi folosiţi doar în cazul în care diferenţele dintre şablon şi regiunea corespunzătoare din imaginea originală sunt suficient de mici (Oi, 1991), (Briechle. 2001), (Tsai, 2002).
  - Algoritmi ce se bazează pe anumiţi descriptori (descriptori Fourier, momente etc.), descriptori globali elementari (arie , perimetru etc.), culoare, semnătură sau textură. Astfel de algoritmi sunt implementaţi în sistemul QBIC (Query Image by Content), prezentat în (Niblack, 1993).

- Algoritmul PCA (Principal Component Analysis - Analiza Componentelor Principale) care implementează o metodă ce se bazează pe potrivirea aspectului obiectului detectat în imaginea analizată, cu un număr mare de imagini ale obiectului țintă (2D sau 3D), prelevate din diverse poziții, într-o etapă anterioară. Autorii acestui algoritm, Murase şi Nayar, prezintă în (Murase, 1995), modelul matematic implementat la baza căruia se află ideea ce presupune că aspectul obiectului, în imagine, este influenţat de formă, reflexie, poziţie în scenă şi iluminare. Metoda PCA, aplicată pentru dezvoltarea algoritmului, provine din statistică, şi este utilizată în acest caz, la compresia imaginilor model.

– Algoritmi ce se bazează pe un model definit cu ajutorul caracteristicilor geometrice (cunoscute) ale obiectului (descriptori de contur):
  - Conturul unui obiect poate fi aproximat cu ajutorul unui poligon. Din această categorie, fac parte algoritmii care aproximează conturul obiectului prin linii drepte (segmente) sau prin curbe (arce de cerc) (Chen, 1996), (Rosin, 1997).
  - Conturul unui obiect poate fi identificat pe bază de graf. Graful reprezintă o mulţime de noduri ce sunt conectaţi printr-o mulţime de arce. Nodurile unui graf reprezintă o trăsătură esenţială a unui obiect (de exemplu o muchie , un colţ) iar arcele descriu relaţia dintre acestea (de exemplu distanţa). Recunoașterea se realizează prin compararea grafului obiectelor detectate în imaginea analizată cu garful modelului învăţat (Li, 1989), (Ersi, 2007).
  - Conturul unui obiect poate fi reprezentat şi unidimensional printr-o caracteristică structurală de tip semnătură. Această semnătură se poate obţine prin reprezentarea conturului în coordonate polare, respectiv prin reprezentarea grafică a distanţelor de la centrul de masă al acestuia până la fiecare pixel din componenţa sa (Gonzalez, 1992), (Aljarrah, 2012).

– Algoritmi ce se bazează pe un model al obiectului construit cu ajutorul unui set finit de puncte caracteristice:
  - Algoritmi bazaţi pe transformata Hough generalizată. Transformata Hough este o metodă de detectare a liniilor drepte, ce ulterior a fost generalizată pentru muchiile unui obiect arbitrar, dacă acesta este cunoscut în prealabil. Ea se aplică, datorită complexităţii, doar obiectelor a căror formă poate fi reprezentată analitic printr-un număr restrâns de parametri. Poate fi folosită în cazul unor imagini cu mai multe obiecte, ce se pot regăsi şi suprapuse. Transformata presupune determinarea pentru fiecare punct a doi parametri. Parametri tuturor punctelor conturului obiectului învăţat sunt introduşi într-un tabel (R- table) care este ulterior utilizat, prin comparare, la operaţia de recunoaștere a obiectului (Ballard, 1981).
  - Algoritmi bazaţi pe distanţa Hausdorff. Distanţă Hausdorff este o metrică neliniară pentru vecinătatea a două seturi de puncte. În cazul algoritmilor de recunoaştere a obiectelor, această distanţă se utilizează ca o măsură a similitudinii între setul de puncte specific modelului învăţat şi setul corespunzător unei regiuni din imaginea analizată. Acest tip de algoritm se utilizează cu precădere la imagini care conţin mai multe obiecte şi al căror contur nu este vizibil în totalitate (parţial ascunse) (Rucklidge, 1997).

2.4. Metode de prelucrare a imaginii

Prelucrarea și analiza imaginilor se referă la ansamblul tehnicilor și metodelor de achiziție, stocare, afișare, modificare și exploatare a informației de natură vizuală din structurile de date bidimensionale (imagini). Prelucrarea propriu-zisă a imaginilor se regăsește, din punct de vedere funcțional, în blocurile de calcul ce transformă imaginea de la intrare într-o altă imagine, cu caracteristici (perceptuale, semantice, structurale) asemănătoare (Jain, 1999), (Pratt, 2001).

Structura unui sistem de prelucrare a imaginilor este ilustrată în figura 2.11.

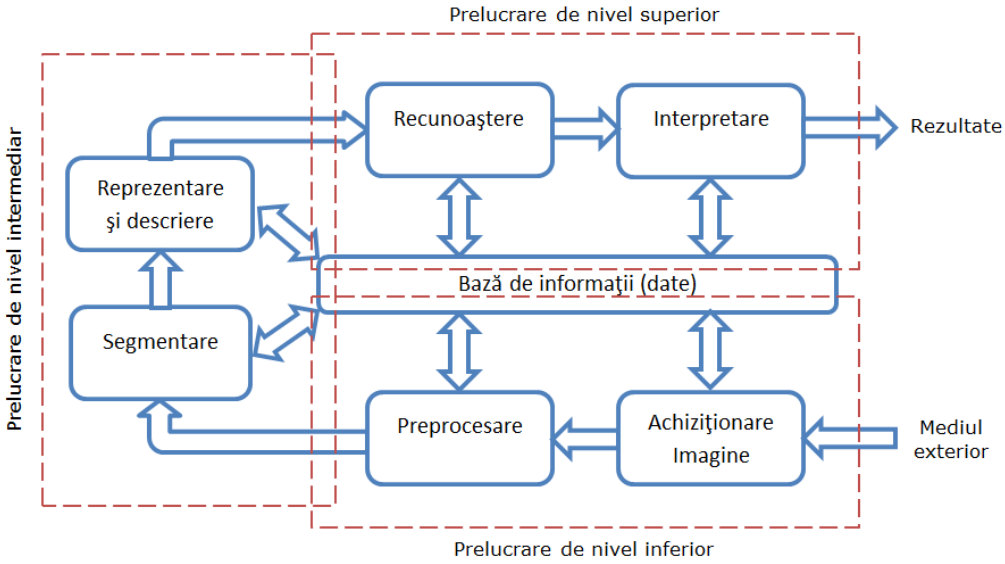


Figura 2. 11: Etapele procesului de prelucrare a imaginilor (adaptată după (Vlaicu, 1997))

În conformitate cu cele prezentate anterior, se poate afirma că procesul de prelucrare a imaginilor este structurat pe trei niveluri (Gonzalez, 2002):

A. Prelucrări de nivel inferior:

Se caracterizează prin faptul că datele de intrare şi de ieşire sunt de tip imagine reprezentate printr-o matrice de puncte. Ca şi principale prelucrări la acest nivel avem:

- Achiziție;
- Îmbunătățirea calității şi amplificarea contrastului;
- Eliminarea paraziților și a zgomotului;

Metodele de prelucrare de nivel scăzut au ca scop principal îmbunătăţirea imaginii în sensul accentuării sau punerii în evidenţă a unor trăsături sau caracteristici ce se regăsesc în imagine, pentru creşterea şanselor de succes ale etapelor ulterioare de prelucrare. Criteriile de evaluare a calităţii unei imagini sunt subiective și specifice aplicaţiei, ceea ce face ca gama metodelor utilizate să fie foarte variată. Metodele folosite în diverşi algoritmi de îmbunătăţire a imaginilor utilizează mai multe tipuri de operaţii care se pot clasifica astfel:

- Operaţii punctuale (modificarea contrastului, transformarea imaginii în negativ, binarizare şi egalizare de histograma);

- Operaţiile locale (eliminarea elementelor parazite din imagine prin eliminarea unei linii sau coloane de pixeli sau diminuarea zgomotului utilizând filtre specifice);
- Operaţiile globale;
- Operaţii geometrice (scalarea, translaţia, reflexia sau rotaţia imaginii);
- Pseudocodarea şi codarea.

B. Prelucrarea de nivel intermediar (mediu):

Se caracterizează prin faptul că datele de intrare sunt de tip imagine, iar cele de ieşire sunt reprezentate sub forma unor structuri simbolice. Principalele operaţii de la acest nivel se referă la procesarea propriu zisă a imaginilor. Cele mai frecvent întâlnite în structura algoritmilor de vedere computerizată, sunt:

- Metode de detectare a conturului. Conturul constituie frontiera obiectelor şi separă obiectele de fundal şi/sau de alte obiecte (în cazul obiectelor suprapuse). Importanţa determinării contururilor este dată de faptul că forma obiectelor este păstrată de contur (Annadurai, 2007), (Petrou, 2010). Cunoaşterea contururilor permite simplificarea imaginii, efectuarea de măsurători cantitative asupra dimensiunii obiectelor şi de măsurători calitative asupra formei. Contururile se evidenţiază prin discontinuitatea niveluri de gri la frontiera obiect/fundal. Discontinuitatea în imagine apare datorită existenţei în scena reală a unei diferenţe notabile de culoare, coeficient de reflexie sau gradient de iluminare.
- Metode de prelucrare bazate pe operaţii morfologice: dilatare, eroziune, umplere, determinare înfăşurătoare, scheletizare;
- Metode bazate pe algoritmi de segmentare a imaginii. Segmentarea este o operaţia de prelucrare a imaginii, ce constă în împărţirea imaginii în zone de interes, după anumite criterii. Această operaţie permite detectarea discontinuităţilor - puncte, linii, muchii (edges), conectarea segmentelor (edge linking), determinarea contururilor (boundaries), filtre globale şi adaptative (thresholdings), histograma; Segmentarea este unul din cei mai importanţi paşi ce trebuie parcurşi în analiza unei imagini. Este o operație dependentă de aplicaţie. În funcție de trăsăturile extrase din imagini, segmentarea se pot clasifica în două categorii fundamentale:
  - Segmentare orientată pe regiuni (pe bază de histogramă, de culoare, de textură);
  - Segmentarea orientată pe contur (extragerea conturului prin metode de gradient şi derivate, metode liniare şi neliniare sau metode bazate pe morfologia matematică);
- Metode bazate pe operaţii de reprezentare şi descriere a formelor: descrierea contururilor, descrierea texturilor, momente statistice invariante, descriptori Fourier, texturi, înlănţuire de coduri, aproximări poligonale, semnături, descriptori topologici.

C. Prelucrări de nivel superior

Constituie etapa în care se operează pe reprezentări simbolice. La acest nivel avem:

- Metode de recunoaştere a formelor, la care rezultatul prelucrării, nu mai este o imagine, ci poate fi o decizie, o descriere sau o interpretare a imaginii iniţiale (Nedevschi, 1998).

Pentru recunoaşterea automată a formelor, se pot utiliza, fie metode matematice, fie metode sintactice. Metodele matematice se grupează, după proprietăţile claselor de forme astfel (Popescu, 2001)

- Metode deterministe;

- Metode sintactice.

În literatura de specialitate, metodele de recunoaștere a formelor se mai regăsesc grupate în patru mari categorii în funcţie de tipul de abordare avut în vedere (Chen, 2006):

- Recunoaşterea prin potrivirea cea mai bună (template matching approach);
- Recunoaşterea prin metode statistice (statistical approach);
- Recunoaşterea cu ajutorul reţelelor neuronale (neural networks approach);
- Recunoaşterea sintactică sau structurală (syntactic or structural approach);

– Metode de interpretare a rezultatelor:
  - algoritmi de determinare a graniţei între clase (*decision boundary)*;
  - algoritmi şi funcţii de clasificare statistică (*optimum statistical Bayes classifier)*;
  - metode neuronale de antrenare şi învăţare (*training by back-propagation)*;
  - metode de derivare şi analiză sintactică a şirurilor de coduri (*scanning)*.

2.5. Calibrarea camerelor video

Aplicațiile din domeniul vederii robotizate au la bază cel puțin o cameră video prevăzută cu un senzor de tip CCD sau CMOS. Principala caracteristica a acestor dispozitive este cea de mapare, respectiv de transformare a informaţiilor 3D din mediul înconjurător în informaţii de natură 2D, şi anume în imagini. În vederea determinării parametrilor specifici acestei transformări este necesară realizarea operaţiei de calibrarea a camerei. Această problemă, include de asemenea, modelarea şi parametrizarea procesului de prelucrare de imagini. În acest scop este necesar să se prezinte aspectele de natură geometrică ale procesului de formare a imaginii pentru a putea face legătura între punctele din scenă şi punctele corespunzătoare din imagine.

2.5.1. Modele geometrice ale camerelor video

Un model de calcul, al unei camere video, în principiu descrie procedura prin care entităţile din spaţiul 3D (puncte, linii etc.) sunt proiectate în imagine şi invers. În literatura de specialitate, există un număr mare de modele, ce definesc camerele video, ce pot fi clasificate după mai multe criterii. Dintre aceste modele sunt menţionate următoarele (Sturm, 2011):

– modele globale (modelul camerei de tip „pinhole” (pinhole model.), modele bazate pe transformata afină (affine models), modele polinomiale bazate pe distorsiuni (polynomial distortion models));
– modele locale (modelul celor două plane (two - plane model));
– modele discrete (modele bazate pe raze (ray - based models)).

A. Modelul „cameră de perspectivă” (perspective camera)

Modelul geometric al proiecţie de perspectivă numit şi modelul camerei de tip „pinhole”, este frecvent utilizat, în domeniul vederii artificiale, pentru rezolvarea unor probleme teoretice cât şi practice (Byröd, 2008). Acest lucru se datorează camerei ideale de tip pinhole ce presupune o apertură punctiformă, fără lentile, şi

care nu introduce distorsiuni în imagine. Modelul este reprezentat schematic în figura 2-12.

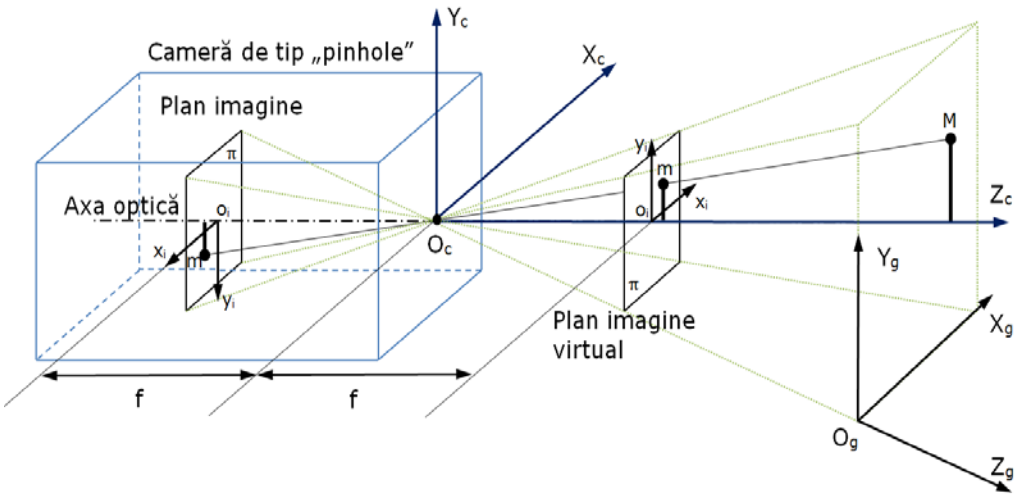


Figura 2. 12: Modelul „cameră de perspectivă”

Modelul constă în:

- Un sistem de referinţă global $O_gX_gY_gZ_g$, ataşat spaţiului de lucru;
- Un punct din spaţiu M de coordonate(X,Y,Z);
- Un sistem de referinţă ataşat camerei ($O_cX_cY_cZ_c$);
- Un plan imagine π;
- Axa optică (linia ce trece prin punctul O_c şi este perpendiculară pe planul π);
- Centrul imaginii o_i numit şi punct principal (aflat la intersecţia axei optice cu planul π);
- Distanţa focală f (reprezintă distanţa dintre punctul O_c şi o_i);
- Un punct m din imagine de coordonate (x, y,–f) aflat la intersecţia liniei ce uneşte punctul M cu O_c (reprezintă proiecţia punctului M în imagine).

Pe baza componentelor prezentate anterior, se poate scrie ecuaţia fundamentală a modelului „cameră de perspectivă”:

$$\begin{pmatrix} x \\ y \end{pmatrix} = \frac{f}{Z}\begin{pmatrix} X \\ Y \end{pmatrix} \qquad (2.1)$$

Se observă:

- Ecuaţia (2.1) nu este liniară datorită raportul $1/Z$;
- Proiecţia punctului M nu este unică, deoarece orice punct de pe axa MO_c are aceiaşi proiecţie;
- Imaginea unui obiect este invers proporţională cu distanţa până la obiect;
- Mărirea distanţei focale f duce la micşorarea câmpului vizual.

B. Modelul proiecţiei ortografice (orthographic projection)

Punctele din spaţiu sunt proiectate cu ajutorul unor raze paralele, ortogonale la planul imagine. În acest caz avem $x=X$ şi $y=Y$.

C. Modelul „camera de perspectivă slabă" (week-perspective camera).

Acest model este o combinație a celorlalte două modele şi presupune (Shimshoni, 1999):

- Valoare medie *Z* a *n* puncte din spaţiu, respectiv $Z_1,\dots Z_n$ este egală cu $\overline{Z}$;
- Punctele din spaţiu sunt proiectate, într-o primă fază, de-a lungul unor raze paralele cu axa optică pe un plan a cărui $Z= \overline{Z}$ astfel încât $x=X$ şi $y=Y$;
- Punctele proiectate sunt supuse, în ce-a de-a doua fază, unei proiecţii de perspectivă cu un factor de scalare $s= f/\overline{Z}$;

Acest model geometric permite transformarea ecuaţia (2.1) într-o ecuaţie liniară:

$$\begin{pmatrix} x \\ y \end{pmatrix} = s \begin{pmatrix} X \\ Y \end{pmatrix} \tag{2.2}$$

În vederea realizării oricărui model de calibrare a camerei trebuie definiţi care sunt parametri acesteia (Zhang, 2008).

5.1.2. Parametrii camerei

Determinarea poziţiei şi orientării unor obiecte în spaţiul de lucru, respectiv stabilirea situării acestora faţă de un sistem de referinţă ataşat spaţiului, se realizează pe baza unui set de ecuaţii. Aceste ecuaţii sunt specifice oricărui algoritm de vedere computerizată, ce este utilizat în aplicaţii de conducere a roboţilor pe bază de imagini, deoarece realizează o legătura între coordonatele anumitor puncte din spaţiul 3D şi coordonatele punctelor omoloage din imagine.

Pentru scrierea acestor ecuaţii este necesar să se realizeze operaţia de calibrare a camerei video, ce presupune determinarea unor caracteristici specifice. Aceste caracteristici, în funcţie de sistemul de referinţă faţă de care se face raportarea se împart (Lenz, 1988), (Savii, 2004), (Ramalingam, 2005):

- caracteristici interne ale camerei cunoscute sub numele de „parametrii intrinseci" ce permit determinării coordonatelor punctelor din imagine faţă de sistemul de referinţă al camerei pe baza coordonatelor pixelilor;
- caracteristici externe camerei cunoscute sub numele de „parametrii extrinseci" ce permit determinarea situării sistemului de coordonate ataşat camerei faţă de un sistem de coordonate din spaţiu 3D (sistemul de coordonate global);

Figura 2.13 ilustrează parametri menţionaţi anterior şi relaţia de legătură dintre sistemul de coordonate ataşat camerei video şi sistemul de coordonate al imagini, respectiv sistemul de coordonate 3D.

A. Parametrii intrinseci

Camerele video utilizate în diverse aplicații de vedere computerizată introduc o varietate de distorsiuni şi aberații, deoarece nu sunt camere perfecte (camere teoretice, ideale tip „pinhole"). Acest aspect conduce la necesitatea determinării parametrilor intrinseci ai camerei.

Parametrii intrinseci se definesc ca „un set de parametri specifici caracteristicilor sistemului optic, geometric şi digital al camerei video" (Hartley, 2003).

Pentru realizarea unor aplicaţii ce necesită măsurători geometrice cu nivel de precizie ridicat, principalii parametri ce trebuie determinaţi sunt cei introduşi de sistemul optic.

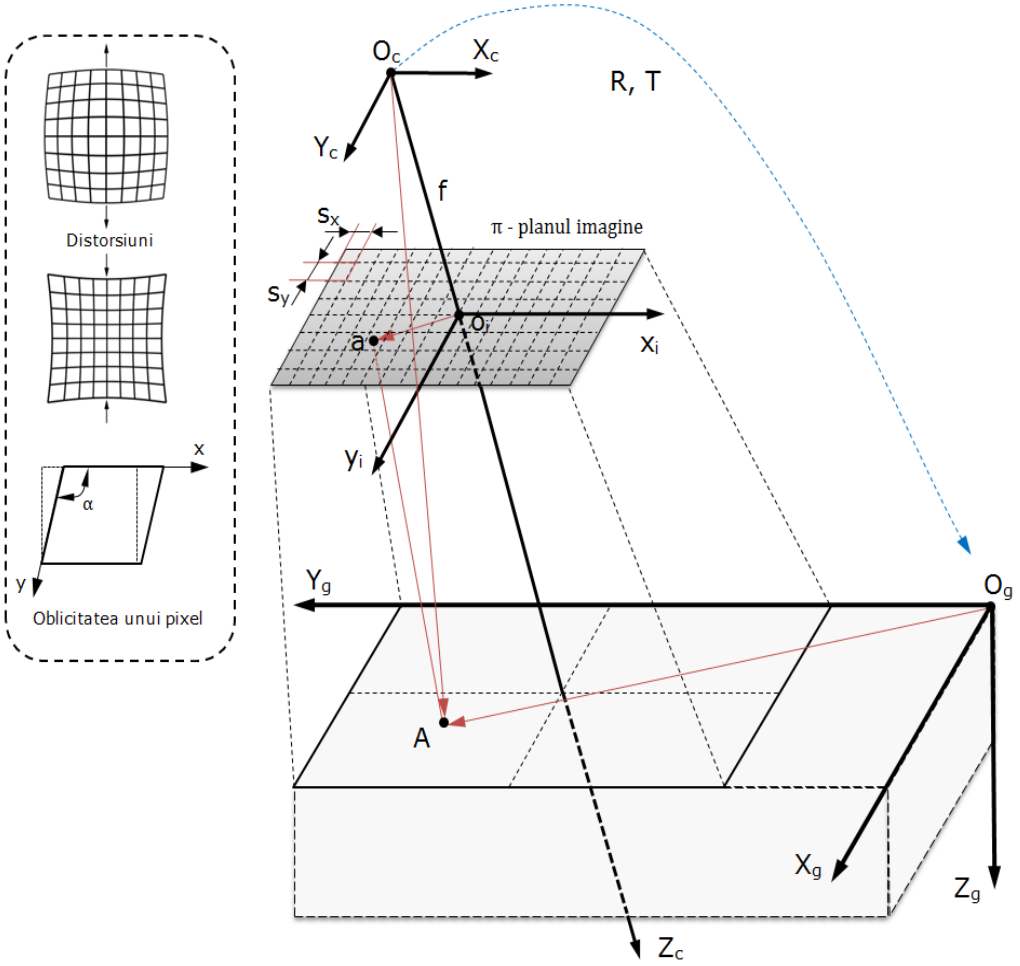


Figura 2. 13: Reprezentarea grafică a parametrilor intrinseci şi extrinseci specifici operaţiei de calibrare a camerei video şi relaţiile de legătură dintre sistemele de referinţă corespunzătoare.

1. parametrii sistemului optic: distorsiunile radiale şi tangenţiale (radial and tangential distortions), distorsiuni de proiecţie (projection distortion) şi unghiul de înclinare sau oblicitatea (skew angle).

Dintre distorsiuni neliniare de mai sus, distorsiunea radială, care este de-a lungul direcție radială de la centrul imaginii, este cea mai severă și mai frecventă (Devernay, 2001). În centrul imaginii valoarea sa este egală cu zero şi creşte pe măsură ce ne deplasăm faţă de acesta.

Îndepărtarea distorsiuni radiale se efectuează frecvent prin aplicarea unui model parametric de distorsiune radială, într-o primă etapă. Acesta permite estimarea coeficienților de distorsiune, care sunt utilizaţi într-o etapă ulterioară la corectarea distorsiuni.

$$r_d = f(r_u)$$
$$r_d^2 = x_d^2 + y_d^2 \qquad (2.3)$$

unde r_u reprezintă raza nedistorsionată, r_d este raza distorsionată iar x_d şi y_d reprezintă coordonatele punctului distorsionat.

Coordonatele punctului nedistorsionat se determină pe baza relației:

$$\begin{aligned} x &= x_d \cdot (1 + k_1 \cdot r^2 + k_2 \cdot r^4) \\ y &= y_d \cdot (1 + k_1 \cdot r^2 + k_2 \cdot r^4) \end{aligned} \tag{2.4}$$

unde k_1 şi k_2 sunt coeficienţii de distorsiune radială.

Gradul de degradare al imagini este proporţional cu valoarea acestor coeficienţi, fapt ce conduce la considerarea lor ca doi parametri intrinseci necesari a fi determinaţi în urma operaţiei de calibrare a camerei.

Un alt parametru este dat de eroarea de asamblare, de către producători, a camerelor video. Această eroare constă în nealinierea senzorului vizual (CCD sau CMOS) cu sistemul optic, respectiv planul imagine nu este ortogonal axei optice.

2. Parametrii necesari transformării coordonatelor camerei (x,y) în coordonatele imaginii – pixeli (x_i, y_i). Aceşti parametri sunt:
   - coordonatele punctului principal, respectiv coordonatele centrului real al imaginii $o_i(x_{oi}, y_{oi})$ în pixeli (intersecţia axei optice cu planul imagine);
   - dimensiunea efectivă a pixelilor în milimetri de-a lungul direcţiei orizontale şi respectiv verticale (s_x, s_y).

$$\begin{aligned} x_i &= \frac{x}{s_x} - x_{oi} \\ y_i &= \frac{y}{s_y} - y_{oi} \end{aligned} \tag{2.5}$$

3. Parametrul geometric de perspectivă: distanţa focală *f* (focal length) în milimetri. Reprezintă distanţa dintre centrul optic al sistemului de lentile al obiectivului camerei video şi planul imagine. Ea poate fi exprimată şi ca doi parametri în pixeli f_x şi f_y (distanţa focală în unităţi de pixeli pe orizontală şi respectiv pe verticală) obţinuţi cu ajutorul factorilor de scalare D_x şi D_y:

$$\begin{aligned} f_x &= f * D_x \\ f_y &= f * D_y \end{aligned} \tag{2.6}$$

Utilizând parametrii prezentaţi anterior, se poate scrie relaţia de legătură între coordonatele punctului *A* în raport cu sistemul de referinţă al imagini $(a(x_i, y_i))$ şi cel al camerei $(A_c(X_c, Y_c, Z_c))$.

$$\begin{bmatrix} x_n \\ y_n \\ 1 \end{bmatrix} = M_{par\_int}^{-1} \begin{bmatrix} x_i \\ y_i \\ 1 \end{bmatrix} \tag{2.7}$$

unde

$$\begin{bmatrix} x_n \\ y_n \end{bmatrix} = \begin{bmatrix} X_c / Z_c \\ Y_c / Z_c \end{bmatrix} \tag{2.8}$$

reprezintă coordonatele normalizate cu 1/Z, iar

$$M_{par\_int} = \begin{bmatrix} f_x & 0 & x_{oi} \\ 0 & f_y & y_{oi} \\ 0 & 0 & 1 \end{bmatrix} \tag{2.9}$$

este matricea internă a camerei ce realizează transformarea dintre sistemul de referinţă al camerei şi sistemul de referinţă al imaginii.

B. Parametrii extrinseci

Parametrii extrinseci se definesc ca „un set de parametri geometrici care permit determinarea situării (poziţia şi orientarea) unui sistem de coordonate 3D, ataşat camerei video, faţă de un sistem de coordonate cunoscut, precum sistemul de referinţă global ataşat spaţiului de lucru" (Chen, 2003).

Transformarea coordonatelor unui punct din sistemul de coordonate global în sistemul de coordonate al camerei şi invers, se face cu ajutorul parametrilor:

- Un vector 3D de translaţie notat cu T care exprimă poziţia relativă dintre originile a două sisteme de coordonate. Această vector poate fi scrisă sub forma:

$$T = \begin{bmatrix} T_x & T_y & T_z \end{bmatrix}^T \tag{2.10}$$

unde T_x, T_y şi T_z sunt translaţiile după axa X, Y si Z.

- O matrice de rotaţie ortogonală de tip 3x3 notată cu R ($R^T \cdot R = R \cdot R^T = I$) care permite suprapunerea axelor corespondente a două sisteme de coordonate. Această matrice poate fi scrisă sub forma:

$$R = \begin{bmatrix} R_1 \\ R_2 \\ R_3 \end{bmatrix} = \begin{bmatrix} r_{11} & r_{12} & r_{13} \\ r_{21} & r_{22} & r_{23} \\ r_{31} & r_{32} & r_{33} \end{bmatrix} \tag{2.11}$$

Utilizând cei doi parametri, se poate scrie relaţia între coordonatele punctului A în raport cu sistemul de referinţă global şi cel al camerei.

$$A_c = R \cdot A + T \tag{2.12}$$

unde $A_c = \begin{bmatrix} X_c & Y_c & Z_c \end{bmatrix}^T$ şi $A = \begin{bmatrix} X_g & Y_g & Z_g \end{bmatrix}^T$.

Pe baza parametrilor definiţi anterior, se pot scrie relaţia de legătură dintre coordonatele în pixeli a unui punct din imagine şi coordonatele punctului echivalent din spaţiul 3D:

$$M_{pr} = M_{par\_int} \cdot \begin{bmatrix} {}^cR_g & | & {}^cT_g \end{bmatrix} \tag{2.13}$$

unde M_{pr} reprezintă matricea de proiecţie, ${}^cR_g$ matricea de rotaţie din sistemul global în sistemul camerei şi ${}^cT_g$ vectorul de translaţie din sistemul global în sistemul camerei.

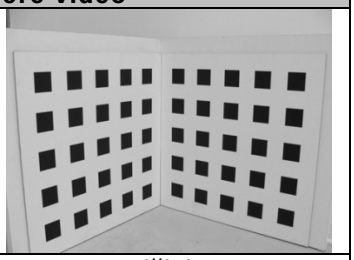
2.5.3. Metode de calibrare a camerei video

O mulţime de cercetători au studiat problema calibrării camerelor fotografice, care se constituie intr-un subiect de cercetare foarte important, începând cu Brown, în vederea obţinerii unui model al camerei cât mai apropiat de cel real şi a unei precizii cât mai bune în cazul parametrilor calculaţi (Martins, 1981), (Wei, 1994), (Wang, 1991), (Beardsley, 1992), (Weng, 1992), (Faugeras, 1996), (Agrawal; Barreto, 2003). Metodele de calibrare pot fi clasificate în două categorii.

Prima categorie se referă la *metode de calibrare off-line* (off-line calibration methods), care include metode tradiţionale de calibrare bazate pe imagini ale unui obiect special de calibrare a cărui structură este cunoscută. Un avantaj important, care se remarcă în cazul utilizării acestor metode constă în faptul că se obţine o precizie ridicată a parametrilor de calibrare. În schimb, dezavantajul constă în faptul că aceste metode nu pot fi aplicate în anumite situaţii cum ar fi cele în care parametrii camerei suferă o serie de modificări în timpul operaţiei sau atunci când se doreşte reconstrucţia unei scene pe baza unor imagini anterior înregistrate (Devernay 2001), (Fung, 2003), (Zhang, 2004), (Cao, 2005), (Datta, 2009).

A doua categorie se referă la *metodele de calibrare on-line* (on-line calibration methods). Cercetătorii care au introdus pentru prima dată aceste metode au fost Faugeras şi colaboratorii săi (Faugeras, 1996), (Maybank, 1992), (Luong, 1997), (Hartley,1997). Aceştia au studiat posibilitatea calibrării unei camere video prin utilizarea corespondenţei dintre punctele prezente în mai multe imagini, fără a cunoaşte vreun detaliu referitor la scenă. Această metodă de calibrare poartă numele de *autocalibrare* (camera self-calibration method) ce permite reconstrucţia unei scene pe baza unor imagini anterior înregistrate. De asemenea, metoda permite calcularea parametrilor camerei în timpul aplicaţiei (Heyden, 1997),(Heikkil, 1997), (Sturm, 1999], (Csurka, 1997), (Cipolla, 1999), (Heikkil, 2000), (Drummond, 2002), (Deutscher, 2002), (Rekleitis, 2005) şi (Hanning, 2007).

În continuare sunt prezentate două metode de calibrare specifice camerelor video. Diferenţa dintre cele două constă în faptul că prima este o metodă mai laborioasă, iar cea de-a doua este o metodă mai rapidă. Cea dintâi ia în calcul doar proprietăţile geometrice în vederea estimării parametrilor camerei prin utilizarea corespondenţei dintre punctele din scenă şi cele din imagine. Cea de-a doua metodă calculează în primă fază matricea de proiecţie, pe baza căreia determină mai apoi parametrii de calibrare (Tsai, 1986), (Faugeras, 1996), (Marita, 2012).

Metoda Tsai de calibrare a unei camere video		
Etape		
1	Se utilizează pentru calibrare un şablon 3D de tip „tablă de şah”; Se achizitioneaza o imagine cu şablonul de calibrare şi se extrag coordonatele „colţurilor” pătratelor şablonului;	
2	Se consideră doua sisteme de referinţă, unul ataşat camerei şi celălalt ataşat şabloului (sistemului global); Se neglijează distorsiunile introduse de obiectiv; Se scriu relaţiile de legatură între coordonatee reale şi coordonatele din imagine;	

$$\begin{bmatrix} X^c \\ Y^c \\ Z^c \end{bmatrix} = R \begin{bmatrix} X^w \\ Y^w \\ Z^w \end{bmatrix} + T \qquad (2.14)$$

$$\begin{aligned} X^C &= r_{11}X^w + r_{12}Y^w + r_{13}Z^w + T_x \\ Y^C &= r_{21}X^w + r_{22}Y^w + r_{23}Z^w + T_y \\ Z^C &= r_{31}X^w + r_{32}Y^w + r_{33}Z^w + T_z \end{aligned} \qquad (2.15)$$

$$\begin{aligned} x_{im} &= -\frac{f}{s_x}\frac{X^c}{Z^c} + o_x \\ y_{im} &= -\frac{f}{s_y}\frac{Y^c}{Z^c} + o_y \end{aligned} \qquad (2.16)$$

Se notează (x_{im}, y_{im}) cu (x, y);
Se identifică parametri intrinseci:

- Distanţa focală *f*;
- Dimensiunea efectivă a pixelului pe orizontală şi verticală (s_x, s_y);
- Coordonatele centrului imaginii (o_x, o_y).

Se exprima parametri sub o alta formă pentru a putea fi calculaţi:

$$\begin{aligned} f_x &= \frac{f}{s_x}; \\ \alpha &= \frac{s_y}{s_x} \end{aligned} \qquad (2.17)$$

Se rescriu relaţiile de legatură;

$$\begin{aligned} x - o_x &= -f_x \frac{r_{11}X^w + r_{12}Y^w + r_{13}Z^w + T_x}{r_{31}X^w + r_{32}Y^w + r_{33}Z^w + T_z} \\ y - o_y &= -f_y \frac{r_{21}X^w + r_{22}Y^w + r_{23}Z^w + T_y}{r_{31}X^w + r_{32}Y^w + r_{33}Z^w + T_z} \end{aligned} \qquad (2.18)$$

$$x_i \cdot f_y(r_{21}X^W + r_{22}Y^W + r_{23}Z^W + T_y) = y_i \cdot f_x(r_{11}X^W + r_{12}Y^{\partial W} + r_{13}Z^W + T_x) \qquad (2.19)$$

Se notează:

$$v_1 := r_{21} \quad v_2 := r_{22} \quad v_3 := r_{23} \quad v_4 := t_y$$
$$v_5 := \alpha \cdot r_{11} \quad v_6 := \alpha \cdot r_{12} \quad v_8 := \alpha \cdot t_x \quad v_7 := \alpha \cdot r_{13}$$

Se rescrie ecuaţia (2.20)

$$x_i X_i^W v_1 + x_i Y_i^W v_2 + x_i Z_i^W v_3 + x_i v_4 - y_i X_i^W v_5 + y_i Y_i^W v_6 + y_i Z_i^W v_7 + y_i v_8 = 0$$

Se utilizează ecuaţia (2.20) rescrisă pentru opt puncte necoplanare;

$$Av = 0 \qquad (2.20)$$

	$A:=\begin{vmatrix} x_1X_1^W & x_1Y_1^W & x_1Z_1^W & x_1 & y_1X_1^W & y_1Y_1^W & y_1Z_1^W & y_1 \\ x_2X_2^W & x_2Y_2^W & x_2Z_2^W & x_2 & y_2X_2^W & y_2Y_2^W & y_2Z_2^W & y_2 \\ x_3X_3^W & x_3Y_3^W & x_3Z_3^W & x_3 & y_3X_3^W & y_3Y_3^W & y_3Z_3^W & y_3 \\ x_4X_4^W & x_4Y_4^W & x_4Z_4^W & x_4 & y_4X_4^W & y_4Y_4^W & y_4Z_4^W & y_4 \\ x_5X_5^W & x_5Y_5^W & x_5Z_5^W & x_5 & y_5X_5^W & y_5Y_5^W & y_5Z_5^W & y_5 \\ x_6X_6^W & x_6Y_6^W & x_6Z_6^W & x_6 & y_6X_6^W & y_6Y_6^W & y_6Z_6^W & y_6 \\ x_7X_7^W & x_7Y_7^W & x_7Z_7^W & x_7 & y_7X_7^W & y_7Y_7^W & y_7Z_7^W & y_7 \\ x_8X_8^W & x_8Y_8^W & x_8Z_8^W & x_8 & y_8X_8^W & y_8Y_8^W & y_8Z_8^W & y_8 \end{vmatrix}$	(2.21)
	Se determină parametri prin descompunerea matricei A cu ajutorul metodei SVD (Singular Value Decomposition – Descompunerea în Valori Singulare);	
	$A=UDV^T$	(2.22)
2	Se calculează factorul de scalare gamma şi raportul de aspect alpha utilizând vectorul soluţie v=ṽ $\bar{v}=\gamma(r_{21},r_{22},r_{23},t_y,\alpha r_{11},\alpha r_{12},\alpha r_{13},\alpha t_x)$ $\sqrt{v_1^2+v_2^2+v_3^2}=\sqrt{\gamma^2(r_{21}^2+r_{22}^2+r_{23}^2}=\lvert\gamma\rvert$ $\sqrt{v_5^2+v_6^2+v_7^2}=\sqrt{\gamma^2(r_{11}^2+r_{12}^2+r_{13}^2}=\alpha\cdot\lvert\gamma\rvert$	(2.23)
3	Se determină matricea de rotaţie şi primii doi termeni ai vectorului de translaţie;	
4	Se impune condiţia de ortogonalitate a matrici de rotaţie şi se determină semnul termenilor acesteia;	
	$x(r_{11}X^W+r_{12}Y^W+r_{13}Z^W+T_x)>0$	(2.24)
5	Se utilizează metoda celor mai mici pătrate (least squares solution) la un sistem de ecuaţii asemănător cu (2.21) pentru N puncte (N=8). Se scrie pentru fiecare punct (xi, yi) relaţia:	
	$x(r_{11}X^W+r_{12}Y^W+r_{13}Z^W+T_x)>0$	(2.25)
6	Se rezolvă sistemul de N ecuaţii lineare;	
	$A\begin{pmatrix}T_z\\f_x\end{pmatrix}=b$	(2.26)
	$A=\begin{pmatrix}x_1 & r_{11}X_1^W+r_{12}Y_1^W+r_{13}Z_1^W+T_x\\ x_2 & r_{11}X_2^W+r_{12}Y_2^W+r_{13}Z_2^W+T_x\\ \vdots & \vdots\\ x_N & r_{11}X_N^W+r_{12}Y_N^W+r_{13}Z_N^W+T_x\end{pmatrix}\quad b=\begin{pmatrix}-x_1(r_{31}X_1^W+r_{32}Y_1^W+r_{33}Z_1^W)\\ \vdots\\ -x_N(r_{31}X_N^W+r_{32}Y_N^W+r_{33}Z_N^W)\end{pmatrix}$	
	$\begin{pmatrix}T_z\\f_x\end{pmatrix}=(A^TA)^{-1}A^Tb$	(2.27)

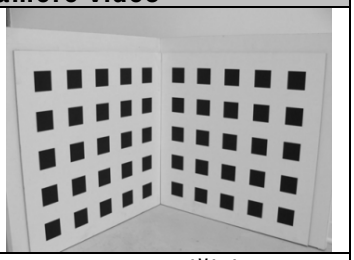
Tabelul 2. 4 Algoritmul metodei Tsai de calibrare a unei camere video. (algoritmul implementat în Matlab se regăseşte la anexa 2).

Analog primei metode şi următoarea metodă are drept scop determinarea paremetrilor intrinseci şi extrinseci ai camerei pe baza unei singure imagini a şablonului de calibrare. Diferenţa faţă de prima metodă constă în faptul că parametrii camerei sunt obţinuţi în urma scrierii mai întâi a matricei de proiecţie, care realizează legătura dintre coordonatele 3D şi cele 2D.

Metoda Faugeras de calibrare a unei camere video

Etape

1 Se utilizează pentru calibrare un şablon 3D de tip „tablă de şah";
Se achizitioneaza o imagine cu şablonul de calibrare şi se extrag coordonatele „colţurilor" pătratelor şablonului;



2 Se consideră doua sisteme de referinţă, unul ataşat camerei şi celălalt ataşat şabloului (sistemului global);
Se scriu relaţiile de legatură dintre coordonatele 3D (X_i^w, Y_i^w, Z_i^w) a unui punct în spaţiu şi coordonatele 2D (x_i, y_i) a proiecţiei sale în planul imagine sub forma unei matrice de proiecţie M de tip 3x4.

$$\begin{pmatrix} u_i \\ v_i \\ w_i \end{pmatrix} = M \begin{pmatrix} X_i^w \\ Y_i^w \\ Z_i^w \\ 1 \end{pmatrix} \qquad (2.28)$$

$$x = \frac{u_i}{w_i} = \frac{m_{11}X_i^w + m_{12}Y_i^w + m_{13}Z_i^w + m_{14}}{m_{31}X_i^w + m_{32}Y_i^w + m_{33}Z_i^w + m_{34}}$$

$$y = \frac{v_i}{w_i} = \frac{m_{21}X_i^w + m_{22}Y_i^w + m_{23}Z_i^w + m_{24}}{m_{31}X_i^w + m_{32}Y_i^w + m_{33}Z_i^w + m_{34}} \qquad (2.29)$$

Se defineşte matricea M până la un factor de scalare arbitrar ce are numai unsprezece parametri independenţi, care la rândul lor se determină pe baza unui sistem liniar omogen;
Se estimează M utilizând metoda celor mai mici pătrate;

$$Am = 0 \qquad (2.30)$$

$$A = \begin{pmatrix} X_1 & Y_1 & Z_1 & 1 & 0 & 0 & 0 & 0 & -x_1X_1 & -x_1Y_1 & -x_1Z_1 & -x_1 \\ 0 & 0 & 0 & 0 & X_1 & Y_1 & Z_1 & 1 & -y_1X_1 & -y_1Y_1 & -y_1Z_1 & -y_1 \\ X_2 & Y_2 & Z_2 & 1 & 0 & 0 & 0 & 0 & -x_2X_2 & -x_2Y_2 & -x_2Z_2 & -x_2 \\ 0 & 0 & 0 & 0 & X_2 & Y_2 & Z_2 & 1 & -y_2X_2 & -y_2Y_2 & -y_2Z_2 & -y_2 \\ \vdots & \dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots & \vdots \\ X_N & Y_N & Z_N & 1 & 0 & 0 & 0 & 0 & -x_NX_N & -x_NY_N & -x_NZ_N & -x_N \\ 0 & 0 & 0 & 0 & X_N & Y_N & Z_N & 1 & -y_NX_N & -y_NY_N & -y_NZ_N & -y_N \end{pmatrix} \qquad (2.31)$$

	$m=[m_{11},m_{12},\ldots,m_{33},m_{34}]^T$	(2.32)
	Deoarece rangul matricei *A* este 11, vectorul *m* poate fi obţinut utilizând metoda descompunerii în valori singulare astfel încât:	
	$A=UDV^T$	(2.33)
2	Se scrie matricea de calibrare *M* utilizând vectorul coloană *m* determinat; Se scriu vectorii 3D pe baza matrici;	
	$M=\begin{bmatrix} -f_x r_{11}+o_x r_{31} & -f_x r_{12}+o_x r_{32} & -f_x r_{13}+o_x r_{33} & -f_x T_x+o_x T_z \\ -f_y r_{21}+o_y r_{31} & -f_y r_{22}+o_y r_{32} & -f_y r_{23}+o_y r_{33} & -f_y T_y+o_y T_z \\ r_{31} & r_{32} & r_{33} & T_z \end{bmatrix}$	(2.34)
	$q_1=[m_{11},m_{12},m_{13}]$ $q_2=[m_{21},m_{22},m_{23}]$ $q_3=[m_{31},m_{32},m_{33}]$ $q_4=[m_{14},m_{24},m_{34}]$	(2.35)
3	Se determină valoarea absolută a factorului de scalare;	
	$M=\gamma M$ $\sqrt{m_{31}^2 m_{32}^2 m_{33}^2}=\lvert\gamma\rvert$ $\sqrt{r_{31}^2 r_{32}^2 r_{33}^2}=\lvert\gamma\rvert$	(2.36)
4	Se normalizează matricea M prin împarţirea fiecărui element cu $\lvert\gamma\rvert$; Se determină primii parametri;	
	$T_z=\sigma m_{34}$ $r_{31}=\sigma m_{31}$ $r_{32}=\sigma m_{32}$ $r_{33}=\sigma m_{33}$	(2.37)
5	Se calculează coordonatele centrului imaginii;	
	$o_x=q_1^T q_3$ $o_y=q_2^T q_3$	(2.38)
6	Se determină distanţa focală exprimată în pixeli	
	$f_x=\sqrt{q_1^T q_1-o_x^2}$ $f_y=\sqrt{q_2^T q_2-o_y^2}$	(2.39)

7	Se determină restul parametrilor extrinseci precum:	
	$r_{11}=\sigma(o_x m_{31}-m_{11})/f_x$ $r_{12}=\sigma(o_x m_{32}-m_{12})/f_x$ $r_{13}=\sigma(o_x m_{33}-m_{13})/f_x$ $r_{21}=\sigma(o_y m_{31}-m_{21})/f_y$ $r_{22}=\sigma(o_y m_{32}-m_{22})/f_y$ $r_{23}=\sigma(o_y m_{33}-m_{23})/f_y$ $T_x= \sigma(o_x T_z-m_{14})/f_x$ $T_y= \sigma(o_y T_z-m_{24})/f_y$	(2.40)

Tabelul 2. 5 Algoritmul metodei Faugeras de calibrare a unei camere video. (algoritmul implementat în Matlab se regăseşte la anexa 3).

Deoarece matricea de rotaţie obţinută nu este o matrice ortogonală, se impune condiţia de ortogonalitate ($RR^T=I$). Acest lucru se realizează prin descompunerea matricei R cu ajutorul metodei SVD astfel încât $R=UDV^T$ după care se înlocuieşte matricea D de tip 3x3 cu matricea identitate I_3. Matricea ce rezultă $R=UIV^T$ este o matrice ortogonală. Cea de-a doua metodă se utilizează atunci când matricea proiecţiilor este suficientă pentru rezolvarea unei probleme de vedere artificială atâta timp cât nu este nevoie de explicitarea individuală a parametrilor. Prima metodă se bazează pe o tehnică des folosită de comunitatea din domeniul „computer vision” şi a fost implementată cu succes în numeroase aplicaţii.

2.6. Aplicaţii robotizate cu vedere artificială

Odată cu scăderea preţului senzorilor vizuali şi respectiv cu creşterea performanţelor unităţilor de calcul, au apărut un număr mare de aplicaţii bazate pe sisteme de vedere artificială a căror aplicabilitate se regăseşte în diverse domenii de activitate precum cel industrial (industria alimentară, industria automobilelor, industria electronică, industria farmaceutică etc.) sau de prestări servicii (medicină, divertisment, logistică etc.) (Obinata & Dutta, 2007).

Atât instituţiile de cercetare publice cât şi departamentele de cercetare a marilor companii din domeniul industrial manifestă un interes deosebit faţă de aplicaţiile de vedere robotizată, datorită beneficiilor economice semnificative pe care acestea le aduc. Din această cauză tehnologia specifică s-a dezvoltat foarte mult într-un timp relativ scurt.

Un avantaj important al utilizării sistemelor de vedere artificială la nivelul roboţilor industriali constă în eliminarea constrângerilor de natură mecanică, fapt ce face ca aplicaţiile industriale să devină tot mai complexe. Un alt avantaj al acestora este acela că un sistem poate fi folosit la o întreagă celulă robotizată astfel încât costul lui se amortizează mai repede.

Aplicaţiile de vedere robotizată în domeniul industrial se pot clasifica conform următoarelor două criterii (Gümüş, 2011):

- În funcţie de potenţialele caracteristici ale obiectului inspectat:
  - Inspecţia calităţii dimensiunilor (dimensiune, formă, poziţie, orientare, aliniere etc.)

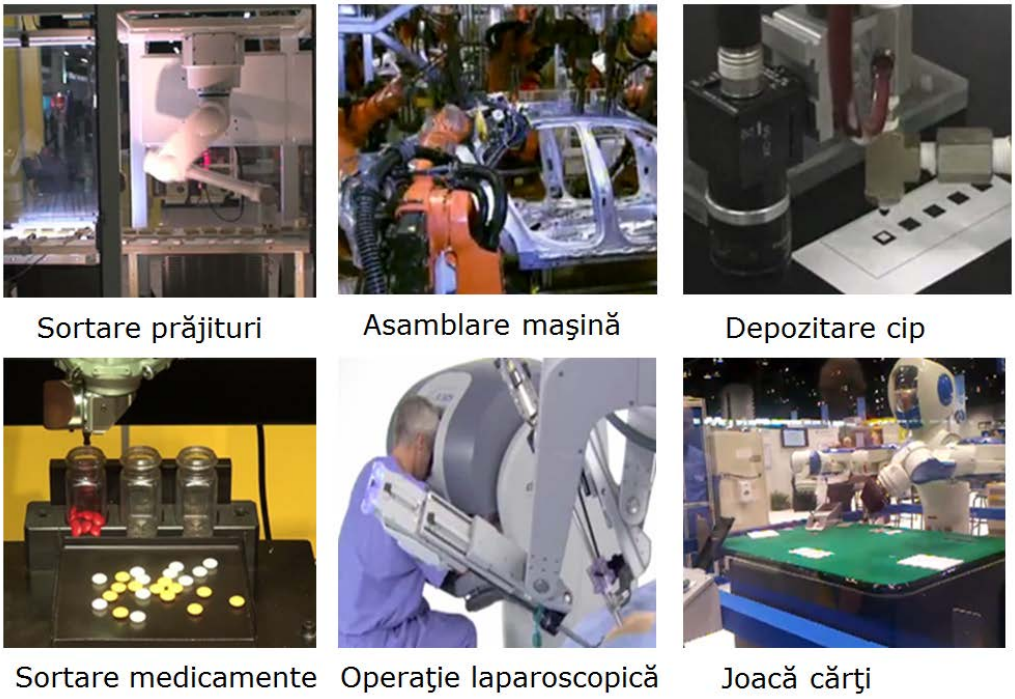


Figura 2. 14: Vederea artificială în diverse domenii de activitate

- Inspecţia calităţii suprafeţei (zgârieturi, crăpături, uzură, finisaj, rugozitate, textură, continuitate, denivelări etc.);
- Inspecţia calităţii structurii (prezenţa/absenţa pieselor componente necesare asamblării precum şuruburi, găuri, nituri, clame şi prezenţa corpurilor străine precum praf, mizerie etc.) ;
- Inspecţia calităţii operaţiei (verificarea funcţionării corecte a produselor inspectate în conformitate cu standardele de fabricaţie).

– În funcţie de gradul de libertate a sistemului:
  - Grad de libertate mare (sisteme ce pot fi extinse din punct de vedere al aplicaţiei – sisteme flexibile);
  - Grad de libertate mic (sisteme utilizate doar pentru aplicaţiile pentru care a fost proiectat – sisteme rigide).

Gradul de libertate al sistemului se găsește în strânsă legătură cu gradul de libertate al obiectului vizat, care la rândul său este dat de caracteristicile acestuia: poziţie, dimensiune geometrică, formă, culoare, textură etc..

– În funcţie de natura operaţiei realizate:
  - Automatizarea producţiei;
  - Eliminarea defectelor;
  - Urmărirea şi identificarea obiectelor;
  - Verificarea ansamblului;
  - Alte aplicaţii specifice;

În continuare sunt prezentate câteva aplicaţii din domeniul industrial unde este folosită vederea artificială pentru controlul roboţilor.

2.6.1. Aplicaţii în industria de automobile

Complexitatea automobilelor din ziua de astăzi a crescut foarte mult, fapt ce a dus la creşterea riscului de apariţie a diverselor erori de-a lungul procesului de fabricaţie. Pentru a evita acest lucru, producătorii din acest domeniu competitiv de activitate au recurs la diverse metode printre care şi la utilizarea vederii artificiale în majoritatea fazelor de producţie.

Firma „IBG Automation GmbH" (Neuenrade, Germany) a dezvoltat un sistem sofisticat de asamblare ce permite poziţionarea şi fixarea roţilor şi de asemenea strângerea şuruburilor la automobile în timp ce acestea se deplasează de-a lungul unei linii de fabricaţie. Sistemul a fost implementat la firma Magna Steyr (Graz, Austria) pentru montarea roţilor unui automobil de tip BMW X3s. Componentele acestui sistem constau în roboţi industriali de tip braţ articulat cu şase grade de libertate de la Kuka prevăzute cu camere video de tip „Matrox Iris GT smart camera" pe care este instalat un algoritmul complex de vedere computerizată realizat cu ajutorul unui soft specializat (Matrox Design Assistant). Acesta permite determinarea poziţiei (x,y), rotaţiei (Rz) şi distanţei (z) centrului jantei faţă de cameră.

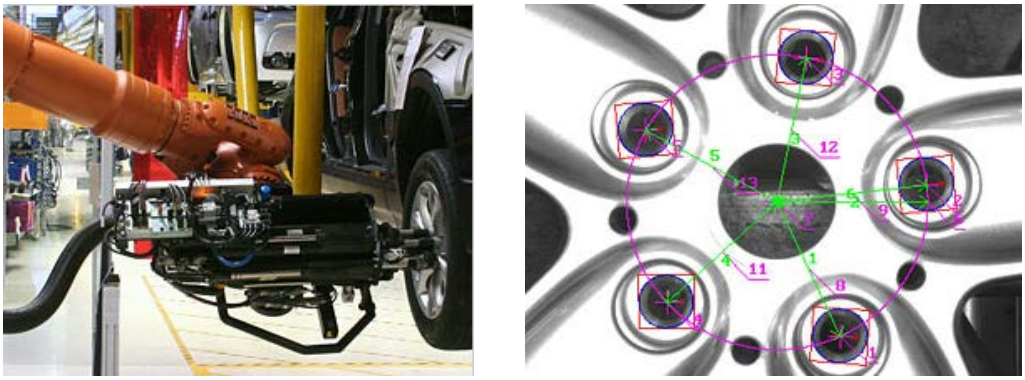


Figura 2. 15: Fixarea roţilor la automobile (AUTOMATICA, 2012)

Un alt exemplu constă într-o operaţie de inspectare a pieselor de tip cap de cilindru utilizând un sistem de vedere artificială într-o celulă robotizată (Kruachottikul 2012). Scopul aplicaţiei este reducera erorilor şi inconsecvenţelor rezultate în urma inspecţiei vizuale manuale prin înlocuirea cu un sistem fiabil de vedere artificială robotizată.

Sistemul de vedere robotizată de tip SOLIMAC conceput pentru inspectarea capetelor de cilindru ale unui automobil este alcătuit din trei staţii ce permit inspectarea şi analiza completă a produselor liniei de fabricaţie şi grupare a acestora în produse ce îndeplinesc sau nu îndeplinesc cerinţele de calitate impuse.

Prima staţie este prevăzută cu două camere video ataşate unei axe de translaţie ce detectează culoarea arcului şi penei fiecărei supape. A doua staţie este compusă dintr-un robot, de care este ataşată o cameră video ce permite o inspectare mai flexibilă a produsului. Acesta realizează poziţionarea precisă a camerei video la un anumit unghi astfel încât să poată fi citit numărul de serie al produsului utilizând metode de recunoaştere de caractere (Optical Character Recognition - OCR), iar ultima staţie, pe baza informaţiilor obţinute anterior, direcţionează produsele.

Figura 2. 16: Sistemului de inspectare SOLIMAC (SOLIMAC, 2012).

Filiala americană a companiei japoneze Ogihara, una dintre cele mai mari companii ce produc şi distribuie componente de înaltă calitate pentru automobile în întreaga lume, a automatizat întregul proces de inspectare dimensională a pieselor şi de asamblare. Acest lucru a fost posibil datorită implementării pe diverşi roboţi industriali a sistemelor de vedere artificială dezvoltate pentru metrologie de firma Hexagon Metrology ce permit prelevarea informaţiilor dimensionale de la obiectele măsurate indiferent de formă, dimensiune sau complexitate (CogniTens, 2009).

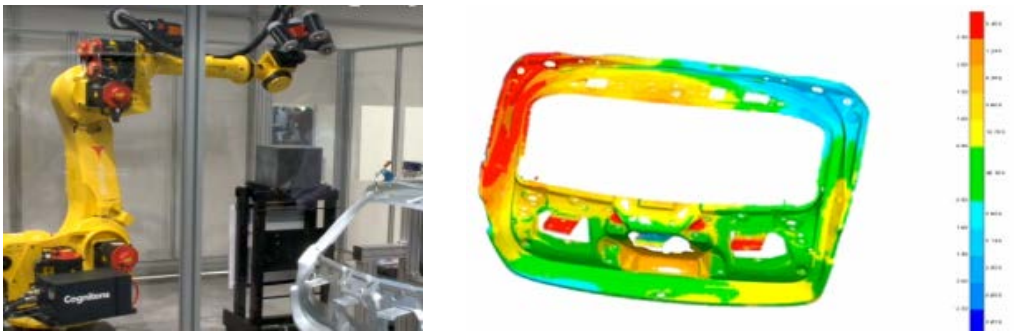


Figura 2. 17: Sistemul de măsurare pe bază de vedere artificială Hexagon(CogniTens, 2009).

Compania internaţională de camioane şi motoare „International Truck and Engine Corporation", a implementat la fabrica sa din Indianapolis, un sistem robotizat cu vedere artificială, ce avea drept scop verificarea amplasării corecte, în timpul asamblării, a punţilor supapei unui motor diesel (Austin 2005). Celula de fabricaţie robotizată deţine un robot cu şase grade de libertate de tip ABB IRB 140 pe care este montată o cameră video. În momentul în care motorul se găsea în spaţiul de lucru al robotului, acesta se deplasa deasupra blocului motor, poziţionându-se în anumite puncte de inspecţie astfel încât camera video să poată preleva imaginile necesare. Algoritmul de procesare permite determinarea poziţiei exacte a celor şaisprezece punți.

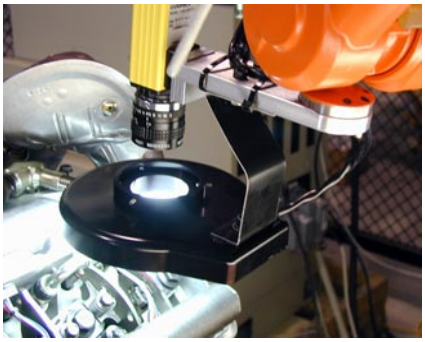


Figura 2. 18: Inspectarea poziţiei punţilor unui motor diesel (VISION 2012).

Alte aplicaţii de vedere robotizată, în industria automobilelor, pot fi de asamblare automată a uşilor şi geamurilor. În aceste cazuri robotul preia piesa vizată şi o aduce într-o anumită poziţie de fixare, iar sistemul de vedere artificială achiziţionează şi analizează imaginile necesare pentru a determina ofset-urile existente. Datele pentru corectarea poziţiei şi orientării sunt ulterior trimise robotului pentru a realiza asamblarea.

Figura 2. 19: Montarea automată a uşilor şi geamurilor unui automobil (Perceptron, 2012).

2.6.2. Aplicaţii în industria alimentară

Industria alimentară, una dintre cele mai importante componente ale economiei unei ţări, beneficiază la rândul ei de avantajele date de vederea robotizată. Aplicaţiile cele mai uzuale sunt de detectare de defecte, manipulare de produse, sortare şi ambalare.

În cadrul unei firme de patiserie din Europa se face sortarea şi ambalarea a optzeci de sortimente de produse. Soluţia adoptata de firma în cauză pentru realizarea operaţiilor, a fost instalarea unei linii robotizate denumită „Astor” compusă din opt roboţi de tip „delta”, conduşi pe bază de vedere artificială care pot executa operaţii de „pick and place” a 140 de unităţi pe minut. Sistemul de vedere artificiala folosit este un sistem de tip „Matrox 4Sight M” ce permitea achiziţionarea şi prelucrarea în timp real a imaginilor cu prăjiturile aflate pe o bandă transportoare.

Algoritmul de vedere utilizat determină poziţia prăjiturii pe bandă şi de asemenea determină dacă forma şi culoarea prăjiturii corespunde calităţii impuse.

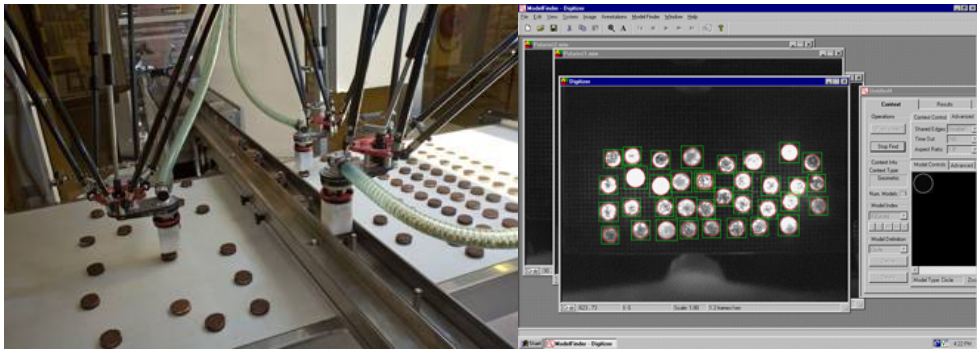


Figura 2. 20: Sortarea prăjiturilor (MATROX, 2012).

Un alt exemplu este cel de manipulare şi depozitare a produselor din carne, proaspete sau congelate, în recipiente specifice. Pentru acest tip aplicaţie un producător a decis să utilizeze roboţi cu configuraţie paralelă, cum este „FlexPicker IRB 360" de la firma ABB, deoarece au viteze de lucru mari şi flexibilitate ridicată. Această alegere este justificată de faptul că feliile de carne sosesc pe o bandă transportoare, la o viteză foarte mare în poziţii aleatoare. Sistemul de vedere artificială, realizează în timp real operaţia de localizare a produsului pe bandă simultan cu operaţia de analiză a calităţii acestuia şi transmite rezultatele obţinute roboţilor. Feliile de carne, ce nu îndeplineau criteriile de calitate, sunt îndepărtate iar celelalte sunt plasate, în caserole, cutii sau tăvi pentru a fi ambalate.

Figura 2. 21: Operaţia de plasarea în caserole a feliilor de carne (VISION 2012).

O altă aplicaţie este cea de dezosare a carcaselor de carne. De obicei această operaţie este realizată manual de către un operator uman. Carcasele de carne, indiferent de tipul de animal, au diferite forme şi dimensiuni, fapt ce face ca automatizarea procesului să fie dificilă, dat fiind gradul de complexitate ridicat al programului necesar a fi implementat pe o maşina sau pe un robot.

Pentru soluţionarea acestei probleme, un grup de cercetători de la institutul de învăţare şi cercetare din Georgia, Statele Unite ale Americi, (Georgia Tech Research Institute - GTRI), au dezvoltat un sistem de vedere robotizată ce permite tăierea şi dezosarea unei găini. Sistemul intitulat "Intelligent Cutting and Deboning System" este compus din doi roboţi şi un sistem de vedere 3D. Un braţ robotizat cu şase grade de libertate manipulează „găina" astfel încât sistemul de vedere 3D să poată preleva imaginile necesare iar celălalt robot cu două grade de libertate să poată realiza operaţiile de tăiere şi de dezosare. Algoritmul estimează poziţia oasele şi ligamentele acesteia, pe baza unor repere identificate în imagini, ce se regăsesc în diferite locaţii în exteriorul păsării (Britton 2011).

Figura 2. 22: Operaţia de tăiere şi dezosare a unei găini (GTRI 2012).

2.6.3. Aplicaţii în industria electronică

Industria electronică este o ramură tânără a economiei, ce derivă din cadrul industriei energetice şi electrotehnice şi care s-a dezvoltat foarte mult într-un timp extrem de scurt. Cercetările şi descoperirile din cadrul acestei industrii precum miniaturizarea componentelor de natură electrică şi electronică au condus la creşterea continuă a cerinţelor pentru inspectarea componentelor, poziţionarea şi localizarea precisă a acestora. Din aceste considerente s-a recurs la vedere computerizată şi vedere robotizată pentru automatizarea operaţiilor de inspectare (determinarea prezenţei sau absenţei componentelor, determinarea montării corecte sau incorecte a pieselor) şi respectiv la automatizarea operaţiilor de asamblare (determinarea situării pieselor, manipularea, inserarea şi lipirea componentelor).

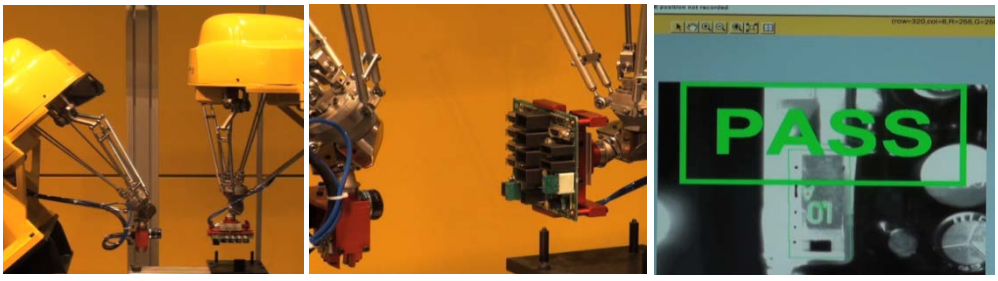


Figura 2. 23: Montarea componentelor electronice pe o placă de calculator (ROBOT, 2012)

Un exemplu de aplicaţie presupune utilizarea vederii robotizate pentru realizarea operaţiei de asamblare a unor componente electronice de dimensiuni mici pe o placă de calculator (o placa de bază).

Celula robotizată concepută pentru această aplicaţie este formată dintr-un sistem de vedere artificială de tip „iRVision" şi doi roboţi industriali. Dat fiind faptul că natura operaţiei de asamblare presupune un grad ridicat de precizie în poziţionare şi viteză mare, roboţii utilizaţi sunt de tip paralel cu şase grade de libertate, de la firma Fanuc (M-1iA).

Aplicaţia în sine constă în manipularea plăcii de calculator de către unul din roboţi, inspectarea ei de către celălalt robot prevăzut cu sistemul de vedere artificială, determinarea locaţiilor unde trebuie amplasate componentele electronice, realizarea operaţiei de asamblare şi inspectarea corectitudini asamblării.

O altă aplicaţie este identificarea şi sortarea de pini pentru asamblarea unor conectori electrici. Compania de roboţi industriali Fanuc a dezvoltat o celulă robotizată compusă dintr-un robot paralel cu şase axe (M-1iA) şi un sistem de vedere artificială compus din două camere de luat vederi. Piesele sunt aduse în zona de lucru a robotului pe o bandă transportoare de alimentare, având poziţii şi orientări aleatoare. Una din camere, montată la nivelul robotului, achiziţionează imagini cu piesele în cauză, iar algoritmul de vedere computerizată implementat, realizat cu ajutorul programului specializat *Fanuc Robotics 2D iRVision*, prelucrează imaginile, determină poziţia acestora şi transmite informaţia necesară prelevării pinilor de către robot. Operaţia de determinare a orientării pinului se realizează cu ajutorul celeilalte camere video, montată sub zona de lucru, în dreptul căreia robotul aduce un capăt al pinului prelevat. În acest fel se determină modalitatea de asamblare a pinului în conector. De asemenea, pe baza imaginilor prelevate de camera doi se determină dacă pinul corespunde cerinţelor de calitate impuse, respectiv dacă pinul este sau nu defect.

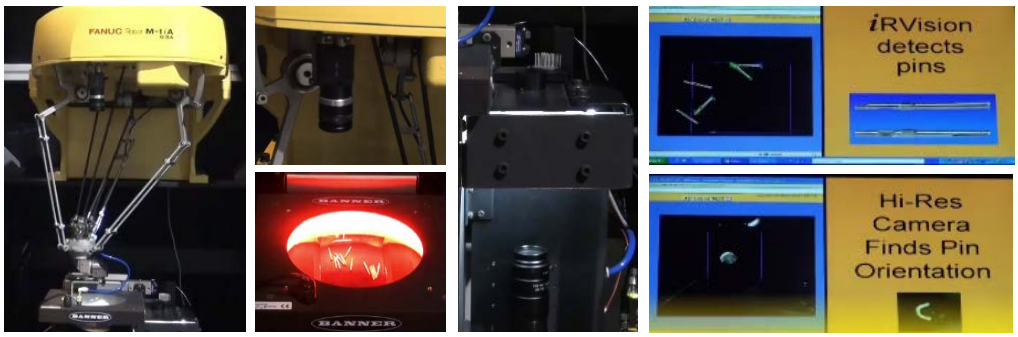


Figura 2. 24: Asamblarea pinilor într-un conector electric (ROBOT, 2012).

2.7. Concluzii

Ţinând cont de multitudinea aplicaţiilor de vedere artificială implementate în diverse domenii de activitate, de numărul semnificativ de cercetări din domeniu vederii robotizate, respectiv din domeniul vederii computerizate şi cel al prelucrării de imagini, se poate afirma că acest capitol subliniază actualitatea tematicii abordate în cadrul prezentei lucrări.

3. OBIECTIVELE ŞI PLANUL DE CERCETARE

3.1. Obiectivele cercetării

Pornind de la analiza cercetărilor actuale din domeniu, teza de doctorat îşi propune ca obiectiv general îmbunătăţirea proceselor de manipulare robotizate pe baza procesării de imagini.

Prezenta cercetare se încadrează conform *Planului Naţional de Cercetare, Dezvoltare şi Inovare, 2007 – 2013*, *PII* în axa prioritară „1 Tehnologia Informației şi Comunicaţii". Direcţia de cercetare „1.4 Inteligenţă artificială, robotica şi sistemele autonome avansate" ce presupune ca tematică de cercetare „7. Dezvoltarea de sisteme inteligente cu autonomie ridicată, inclusiv roboţi autonomi".

Cercetările efectuate de-a lungul studiilor doctorale au fost focalizate, în principal, pe îndeplinirea unui obiectiv principal care constă în dezvoltarea și implementarea unor algoritmi de vedere computerizată destinaţi conducerii automate şi semi-automate a roboţilor.

Obiective derivate sunt:

A. Conceperea şi dezvoltarea unui algoritm de prelucrare de imagini, care să poată fi utilizat într-un proces semi-automatizat.
B. Dezvoltarea unor aplicații de vedere robotizată (robot vision) prin elaborarea unor algoritmi de vedere computerizată (computer vision) pe baza metodelor existente de prelucrare şi procesare a imaginilor cu scopul de recunoaștere a unor obiecte industriale 2D (de tip placă);
C. Îmbunătățirea metodelor existente de procesare a imaginilor şi elaborarea unor algoritmi pentru recunoașterea obiectelor de tip 3D;
D. Dezvoltarea şi implementarea unui algoritm de vedere computerizată într-o celulă robotizată în vederea optimizării prinderii şi manipulării unui obiect de tip industrial.

Pentru atingerea acestor obiective a fost necesară întreprinderea următoarelor direcţii de acţiune:

1. Realizarea unei cercetări extinse cu privire la metodele de prelucrare a imaginilor și recunoaștere a formelor ce sunt utilizate în cadrul algoritmilor de vedere computerizată;
2. Analiza diverselor tehnici, existente la ora actuală, de conducere a sistemelor robotizate pe baza algoritmilor de vedere computerizată;
3. Studierea aplicaţiilor de vedere robotizată existente la ora actuală în domeniul industrial şi identificarea posibilelor operaţii ce pot fi îmbunătăţite sau dezvoltate.
4. Identificarea unei modalităţi de determinare a poziţiei robotului în spaţiul de lucru, pe baza prelucrării de imagini;
5. Detectarea unor defecte de suprafaţă, ale unei structuri, pe bazaă de imagini, pentru determinarea efectelor acestora asupra integrităţii ansamblului.
6. Identificarea obiectelor solide 2D (obiecte de tip placă la care cea de-a treia mărime este neglijabilă în raport cu celelalte două) din imaginile prelevate

de sistemul de vedere artificial pentru realizarea de către robot a operaţiei de manipulare;

7. Identificarea obiectelor 3D din imaginile achiziţionate de la camera video ce este ataşată unui robot în vederea îndeplinirii funcţiei de manipulare;
8. Determinarea unei modalităţi de optimizare a operaţiei de manipularea a pieselor industriale de către robot şi conceperea unui model matematic care să fie implementat într-o aplicaţie de vedere robotizată.

3.2. Planul de activitate

Obiective	Activităţi	Rezultate
Obiectivul A Obiectivul B Obiectivul C Obiectivul D	Activitatea 1	• Rezultatele obţinute au fost valorificate prin întocmirea referatului cu titlul *"Stadiul actual al metodelor de prelucrare a imaginilor în timp real"*.
	Activitatea 2 Activitatea 3	• Informaţiile acumulate de-a lungul acestor activităţi au fost concretizate în referatul cu titlul *"Conducerea sistemelor mobile pe baza vederii artificiale".*
	Activitatea 4	• Au fost implementaţi în Matlab doi algoritmi de calibrare a camerelor video, ce se regăsesc în literatura de specialitate.
Obiectivul A	Activitatea 5	• A fos dezvoltată în Matlab o aplicaţie de identificare a defectelor de suprafaţă, ce pot să apară la nivelul faţadelor clădirilor.
Obiectivul B	Activitatea 6	• S-a proiectat în LabView două aplicaţii de recunoaştere a obiectelor 2D. • A fost dezvoltată o aplicaţie de detectare, recunoaştere a unor obiecte colorate. în vederea manipulării robotizate
Obiectivul C	Activitatea 7	• Au fost elaborate trei aplicaţii industriale de tip „pick and place" în Matlab care permit detectarea, recunoaşterea, identificarea, sortarea şi manipularea unor obiecte 3D.
Obiectivul D	Activitatea 8	• A fost implementă într-o aplicaţie de vedere robotizată un algoritm de optimizare a prinderii obiectelor între bacurile dispozitivului de prehensare.

4. APLICAŢIE DE INSPECTARE A SUPRAFEŢELOR

În cadrul acestui capitol este prezentată o aplicaţie de prelucrare de imagini, ce poate fi folosită la îmbunătăţirea operaţiei de inspectare a făţadelor clădirilor, prin automatizarea procesului de prelucrare a informaţiei utile.

4.1. Introducere

Scopul aplicaţiei: Identificarea defectelor ce apar la nivelul structurii faţadelor clădirilor în vederea determinării efectelor acestora asupra integrităţii clădirii.

Orice clădire existentă la ora actuala se deteriorează în timp din cauza diverşilor factori naturali (dilatarea şi contracţia solului, cutremur, ploi, ninsoare, vânt şi schimbări de temperatură) sau artificiali (vibraţii produse de om, supraîncărcare şi intervenţii distructive la nivelul structurii de rezistenţă). Plecându-se de la acest aspect, ce presupune costuri de restaurare ridicate, apare nevoia de concepere a unor metode non-distructive de monitorizare a „sănătăţii" structurii clădirilor. Metodele existente la ora actuală sunt realizate la înălţime, de personal calificat şi sunt periculoase, laborioase şi imprecise datorita erorilor ce pot să apară ca efect al oboselii operatorului uman. De asemenea, exactitatea rezultatelor prezentate de raportul de diagnosticare diferă în funcţie de specialist datorita factorului subiectiv. Din acest motiv a fost dezvoltat un sistem mai eficient de evaluare a defectelor ce pot să apară la nivelul faţadelor clădirilor (Pop, 2010b).

4.2. Sistemul de inspectare

În vederea realizării unei aplicaţiii de identificare a defectelor de tip crăpatură, ce apar la nivelul faţadelor clădirilor, este necesară utilizarea unor sisteme mecanizate sau robotizate de prelevare controlată a imaginilor corespunzătoare. Două variante posibile, de asfel de sisteme, ce ar putea fi utilizate, sunt ilustrate în figura 4.1.

Paşii necesari a fi parcurşi pentru realizarea acestei aplicatii sunt prezentaţi succint după cum urmează:

Pasul 1. Prelevarea imaginilor necesare realizării operaţiei de mapare a faţadei unei clădiri. Deplasarea camerei video într-un plan paralel cu faţada cladirii se realizează în mod automat şi controlat;

Pasul 2. Îmbunătăţirea imaginilor prelevate prin aplicarea unor metode de procesare a imaginii;

Pasul 3. Stabilirea corespondenţei între imaginile prelevate şi porţiunile din faţadă;

Pasul 4. Analizarea fiecărei imagini achiziţionate;

Pasul 5. Determinarea fisurilor faţadei conţinute într-o singură imagine;

Pasul 6. Concatenarea imaginilor cu eliminarea zonelor comune;

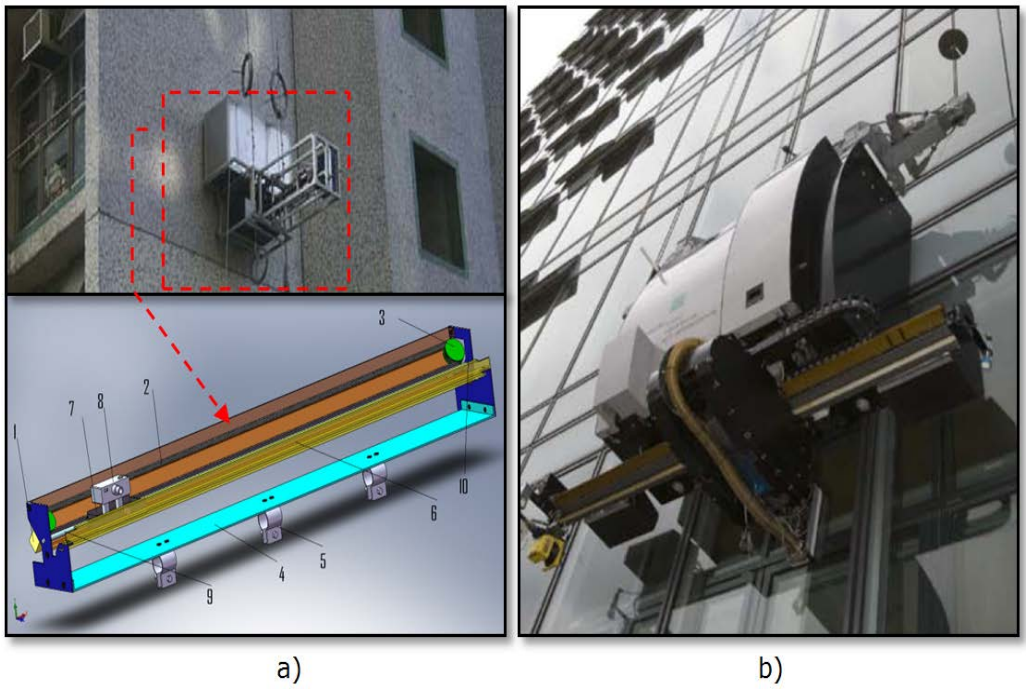


Figura 4. 1: Posibile sisteme de prelevare imagini a) Dispozitiv ghidare cameră, montat pe o nacelă; b) Robotu cartezian SIRIUSc (Elkmann, 2008)

Pasul 7. Determinarea fisurilor din imaginile combinate;

4.3. Aplicaţia în Matlab

Partea software, ce ţine de detectarea şi analiza defectelor, a fost realizată pentru a automatiza procesul de analiză şi diagnosticare a defectelor ce apar la nivelul faţadelor clădirilor. Acest program, respectiv algoritmul de procesare de imagini (anexa 4), a fost dezvoltat cu ajutorul mediul de lucru Matlab, şi permite detectarea defectelor de tip crăpătură şi măsurarea lungimii şi unghiului de înclinare faţă de orizontală pe baza imaginilor prelevate de sistemele anterior menţionat.

Pentru realizarea detectării şi măsurării crăpăturilor, imaginile faţadei clădirii ce au fost achiziţionate au trebuit să fie supuse diverselor operaţii de prelucrare precum: corectarea defectelor, îmbunătăţirea contrastului şi luminozităţii respectiv îmbunătăţirea vizibilităţii anumitor entităţii din imagine, delimitarea obiectelor faţă de fundal etc., operaţii ce au fost integrate în algoritmul programului de prelucrare de imagini.

În vederea îndeplinirii scopului aplicaţiei şi utilizării cu uşurinţa de către un operator uman, algoritmul de prelucrare este prevăzut cu o interfaţă grafică de tip GUI realizată tot în mediul de lucru Matlab (figura 4.2). Interfaţa deţine un număr de butoane cu diferite funcţii, casete text în care se pot introduce s-au afişa valori numerice, meniuri pentru modificarea caracteristicilor, toate fiind grupate în două panouri ce delimitează cei doi paşi necesari a fi parcurşi pentru obţinerea

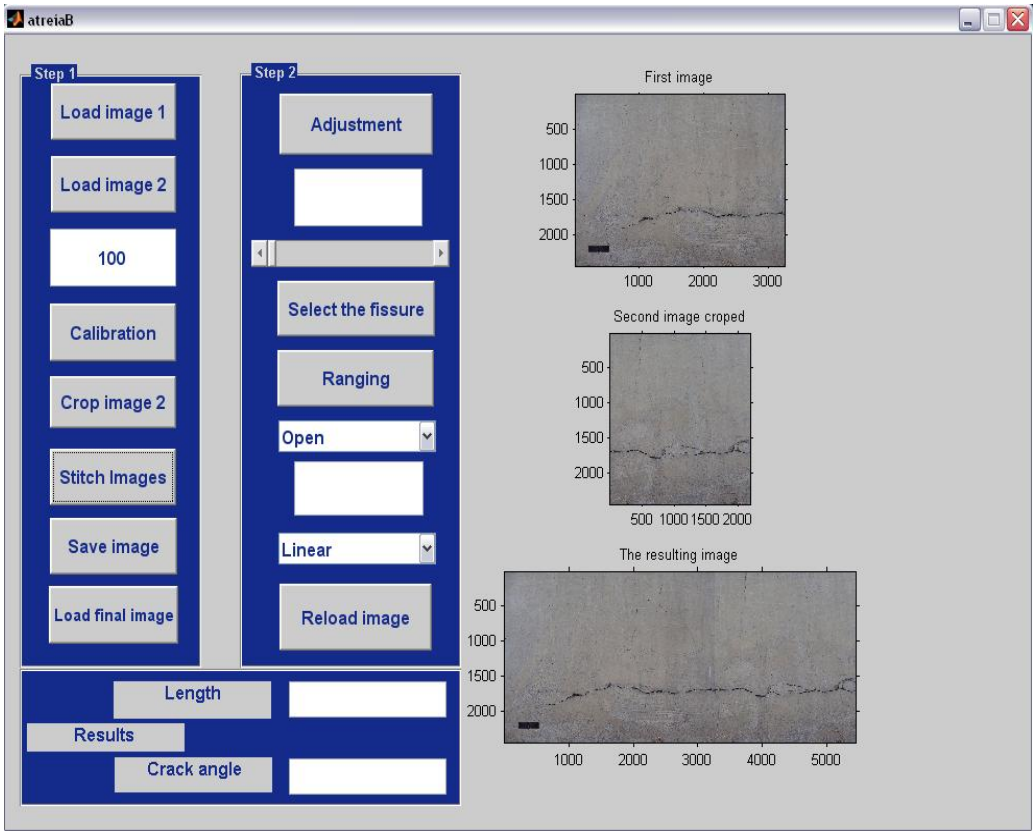


Figura 4. 2: Interfaţa grafică

rezultatelor scontate şi de asemenea este prevăzută şi cu ferestre pentru afişarea rezultatelor.

Interfaţa prezintă două butoane de încărcare de imagini în interfaţa grafică (Load image 1 şi Load image 2) cu scopul de a fi prelucrate ulterior. Butoanele permit încărcarea fişierelor de tip jpg., care în urma operaţiei de achiziţie sunt stocate pe calculator.

Următorul buton (Calibration) împreună cu caseta text asociată, permit realizarea unei aşa numite calibrări a sistemului. Prin calibrare, în acest caz se înţelege relaţia de legătură dintre unitatea de măsură din imagine (pixel) şi unitatea de măsură din realitate (milimetri). Pentru a se putea realiza această operaţie este necesar ca sistemul de care se ataşază camera video să execute o mişcare lineară şi să menţină planul obiect, respectiv faţada clădirii, paralel cu planul imagine (senzorul video). De asemenea, camera video, este necesar să fie prevăzută cu un etalon de tip riglă, ce intra în contact cu planul obiect şi să fie acţionată doar în momentul când se afla la capăt de cursă în poziţie staţionară. Prin urmare, în caseta text se introduce valoarea în milimetri a etalonului ce este prelevat împreună cu fiecare imagine. Prin acest procedeu se asociază o anumită valoare în milimetri unui anumit număr de pixeli.

Utilizând relaţiei dintre pixel şi milimetru obţinută anterior, pe baza lungimii cursei şi pe baza dimensiunii câmpului vizual, algoritmul poate determina cât din a doua imagine se suprapune peste prima. De aceea butonul (Crop image 2) în urma acţionării elimină din a doua imagine surplusul de informaţie.

Următorul buton din primul panel (Stitch images) realizează lipirea celor două imagini obţinute, astfel încât se obţine o singură imagine ce deţinea întreaga informaţie din cele două.

Iar ultimele butoane (Save image şi Load image) realizează salvarea imaginii obţinute pe unitatea de stocare şi încărcarea ei în interfaţa grafică pentru a putea fi ulterior prelucrată.

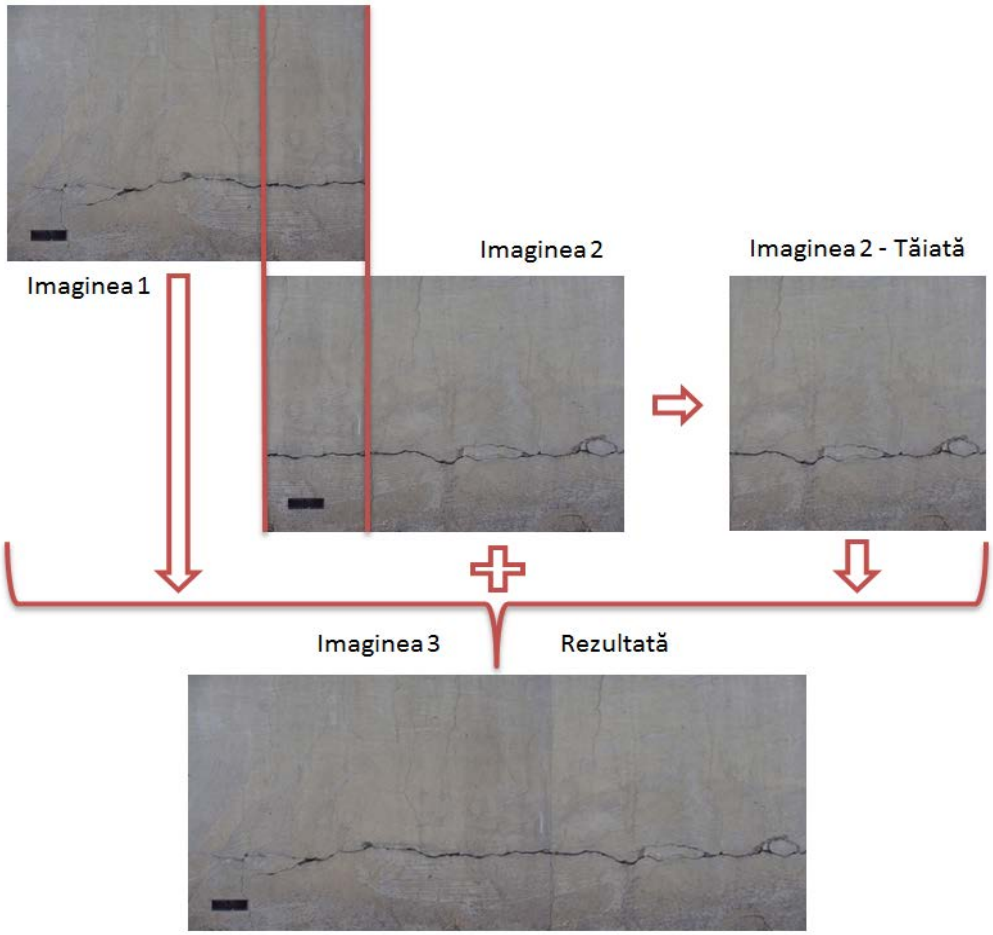


Figura 4. 3: Imaginea obţinută după parcurgerea primului pas

În cazul în care crăpătura ce se doreşte a fi analizată este mai mare şi se regăseşte în mai mult de două imagini, operaţiile de la pasul 1 se repetă. Se încarcă imaginea rezultată ca imaginea 1 iar imaginea următoare prelevată de sistemul video ca fiind imaginea 2. Se repetă operaţiile de la pasul unu până când imaginea rezultată conţine întreaga crăpătură.

Pasul următor al interfeţei permite utilizatorului să proceseze imaginea rezultată în vederea obţinerii informaţiei necesare despre crăpătura analizată. Prima operaţie a acestui pas constă în îmbunătăţirea imaginii pentru a evidenţia cât mai bine crăpătura. Butonul care realizează acest lucru este „Adjust Contrast" ce permite practic modificarea contrastului şi luminozităţii imaginii pe baza histogramei. Această operaţie este opţională deoarece este condiţionată de imaginea rezultată.

Următoarea operaţie de procesare a imagini este cea de binarizare şi se realizează cu ajutorul unei bare de rulare. Operaţia este necesară pentru evidenţierea crăpăturii şi presupune modificarea valorii fiecărui pixel astfel încât imaginea în niveluri de gri (0 corespunde pentru negru, iar 255 reprezintă intensitatea maximă sau alb) se transformă în imagine binară (0 corespunde pentru negru şi 1 pentru alb). Modificarea se realizează manual, iar nivelul de prag este indicat într-o casetă text.

Butonul următor (Select the fissure) permite selectarea unei zone ce conţine crăpătura ce urmează a fi analizată. Această operaţie este de asemenea opţională şi se realizează doar în cazul în care imaginea conţine mai multe crăpături şi se doreşte analizarea uneia şi eliminarea celorlalte. Practic se realizează extragere din imaginea originala a zonei ce se doreşte a fi analizată. Pentru revenirea la imaginea iniţială se utilizează butonul de reîncărcare a imaginii (Reload image).

Panoul acestui pas este prevăzut şi cu două meniuri de tip pop-up. Primul permite selectarea operaţiei morfologice ce urmează a fi utilizată (operaţie de tip open sau close). Acest control realizează o curăţare a imaginii astfel încât crăpătura să fie cât mai bine evidenţiată. Cel de-al doilea permite selectarea tipului de crăpătură ce urmează a fi analizată. Aceasta poate fi liniară, bifurcată sau curbă. În funcţie de tipul de crăpătură aleasă, selectarea în imagine se realizează în mod diferit. Selectarea unei crăpături liniare, în program, se face prin selectarea celor două extremităţi ale acesteia. Deoarece precizia necesară în acest caz este mică s-a considerat că este suficient să se aproximeze lungimea printr-o dreaptă. Lungimea crăpăturii şi unghiul corespunzător sunt afişate în două casete text aflate în colţul stânga jos al interfeţei. Dacă în imagine există o crăpătură bifurcată atunci selectarea se face de la începutul crăpăturii până la bifurcare, după care se selectează fiecare ramură a bifurcaţiei dintr-un capăt în altul. În cazul unor crăpături curbe se selectează câteva puncte de pe curbă şi capetele crăpăturii.

Rezultatele acestei aplicaţii în urma procesării şi analizei crăpături sunt lungimea crăpăturii şi unghiul de înclinare al acesteia.

Dacă în cazul crăpăturii liniare rezultatele sunt obţinute imediat, în cazul crăpăturii bifurcate lungimea acesteia este suma totală a tuturor segmentelor crăpăturii, iar unghiul rezultat, care este afişat în caseta text, este unghiul dat de ramura cu lungimea cea mai mare.

În cazul crăpăturilor curbe lungimea este dată de suma segmentelor care formează curba iar valoarea unghiului de înclinare faţă de orizontală nu este afişat deoarece nu există un segment mai mare care să fie reprezentativ pentru întreaga crăpătură.

Rezultatul obţinut în cazul analizei unei crăpături liniare, cu o valoare a pragului de binarizare de 90 şi utilizând operaţia morfologică open este reprezentat în figura 4.4.

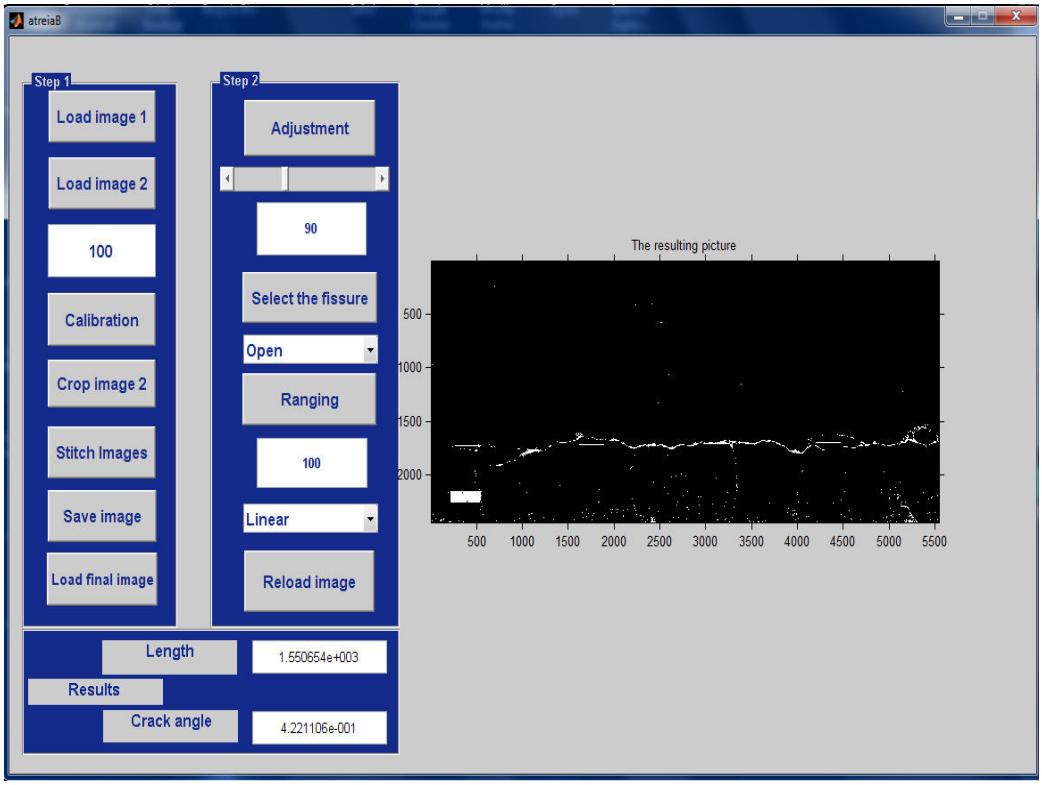


Figura 4. 4: Interfaţa grafică cu rezultatul obţinut

4.4. Concluzii şi rezultate

Aplicaţia de inspectarea a faţadelor clădirilor, prezentată în acest capitol reprezintă o îmbunătăţire a sistemelor existente de detectare a fisurilor de suprafaţă, deoarece folosește instrumente de procesare a imaginii, ce permit evitarea erorilor umane de detecție precum:

- identificări eronate de fisuri (crăpături);
- detectare lentă;
- rezultate subiective;
- ineficienţa gestionării datelor;

Rezultatele obţinute în urma realizării acestei aplicaţii de prelucrare de imagini, au fost diseminate într-o lucrare ştiinţifică, ce a fost publicată şi indexata într-o bază de date internaţională (Pop, 2010b).

5. PROIECTAREA DE APLICAŢII ÎN LABVIEW PENTRU PROCESAREA IMAGINILOR ŞI CONTROLUL ROBOŢILOR

În cadrul acestui capitol sunt prezentate două aplicaţii. Prima aplicaţie este de recunoaștere şi clasificare a unor obiecte de tip ţeavă. Cea de-a doua aplicaţie este de conducere a unui robot cartezian pe baza unei traiectorii generate cu ajutorul unor informaţii extrase din imagini prelevate de la un obiect 2D aflat în spaţiul de lucru al robotului. Aplicaţiile prezentate, au fost realizate pe parcursul unui stagiu de cercetare efectuat, pe o perioadă de 6 luni, la Universitatea de Studi din Udine, Italia (Universita degli Studi di Udine, Facolta di Ingegneria, Dipartimento di Ingegneria Elettrica, Gestionale e Meccanica).

5.1. Aplicaţie de identificare a tipului de obiecte

Scopul aplicaţiei: Detectarea, recunoaşterea şi clasificarea unor obiecte din imagini pe baza unui algoritm de vedere computerizată, dezvoltat pentru îmbunătăţirea proceselor industriale de identificare a diverselor obiecte.

Problema de recunoaştere a obiectelor pe bază de imagini este un subiect de vedere computerizată care a fost studiat de o lunga perioadă de timp, dar care prezintă în continuare un interes deoarece continuă să fie tratat în actualele cercetări din domeniu.

Aplicaţia constă în identificarea unor obiecte de tip ţeavă din imaginile achiziţionate de o cameră video ce este montată într-o poziţie fixă, pe un trepied, faţă de planul obiect (Pop, 2011b). Pentru realizarea acestei aplicaţii s-a utilizat standul experimental din figura 5.1.

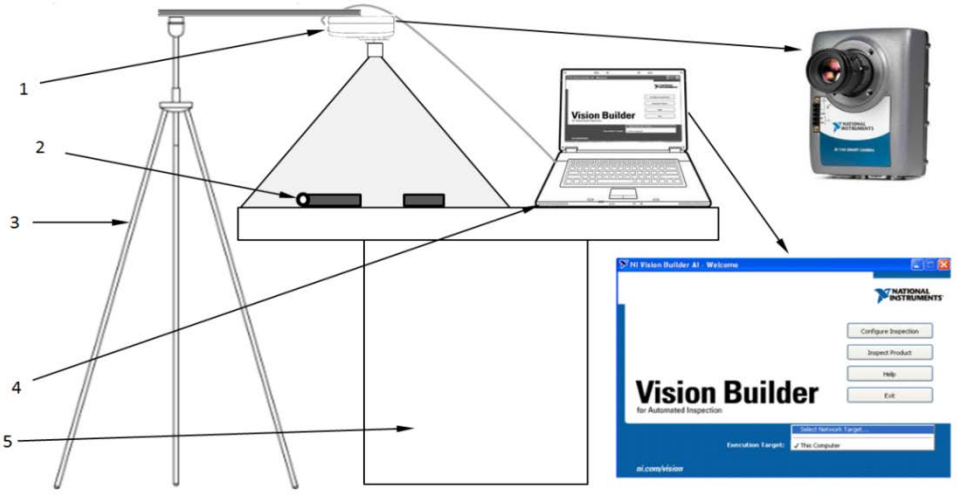


Figura 5. 1: Stand experimental: 1) cameră video inteligentă; 2) piese; 3) trepied; 4) calculator prelucrare date; 5) masă de lucru.

Echipamentele utilizate pentru realizarea acestei aplicaţii sunt:

- Sistem video, concretizat printr-o camera video NI 1741 de tip „smart camera" de la National Instruments;
- Piese de lucru 2D (ce-a dea treia dimensiune este constanta şi se neglijează) de tip „ţeavă";
- Sistem de calcul, pe care rulează un program specializat (Vision Builder for Automated Inspection);

Acest program, utilizat în aplicaţii industriale de vedere computerizată, a fost conceput de National Instruments, special pentru realizarea operaţilor de inspecţie vizuală, de detectare a prezenţei unor componente, de detectare a defectelor, de realizare a unor operaţii de numărare şi de măsurare.

Cu ajutorul acestui program, a fost dezvoltat de autor algoritmul de vedere computerizată, corespunzător aplicaţiei de faţă, algoritm la baza căruia se află o diagramă de stare, ce este ilustrată în figura 5.2.

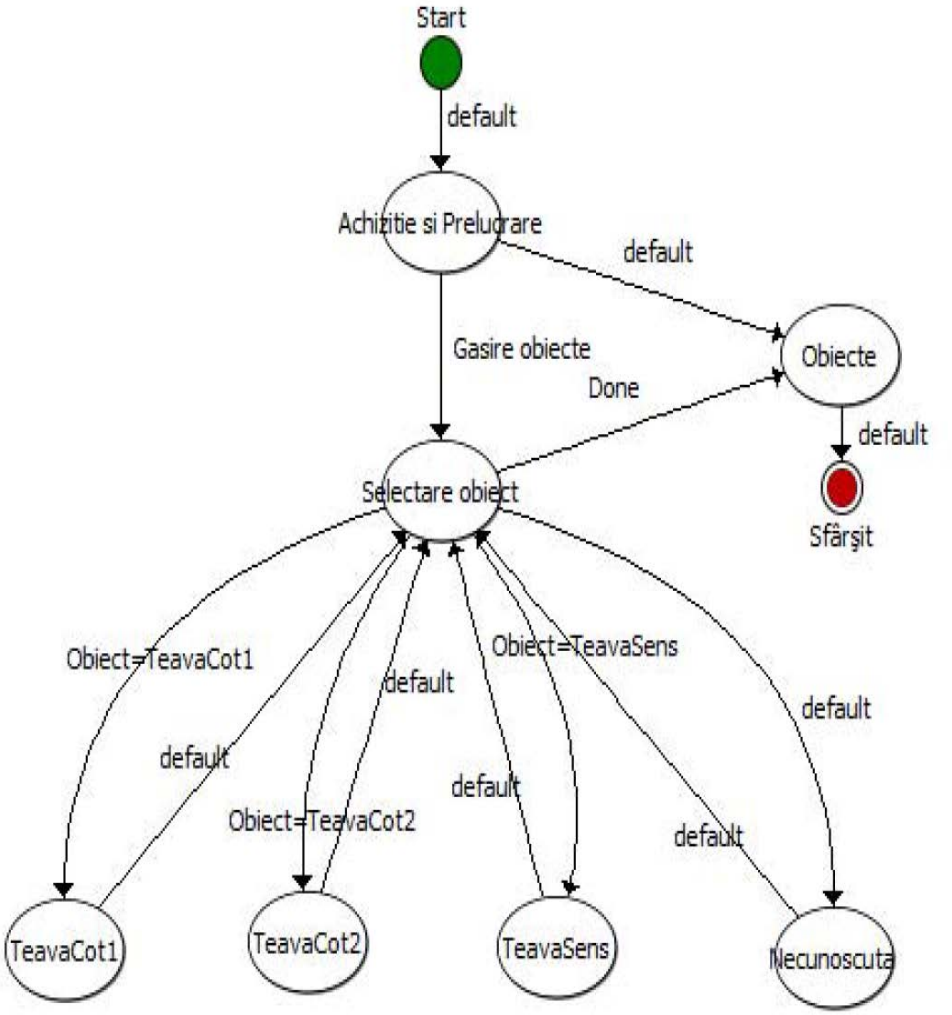


Figura 5. 2: Diagrama de stare a aplicaţiei

Diagrama este compusă din şapte stări şi tranziţiile aferente acestora. Fiecare stare a acestei diagrame presupune o rutină de operaţii de procesare de imagini, operaţii ce sunt ilustrate în figura 5.3.

Pentru realizarea acestei aplicaţii de vedere computerizată s-a ţinut cont de următoarele premise:

- Date de intrare:
  - În spaţiul de lucru se pot găsi mai multe obiecte necunoscute;
  - Şabloanele necesare operaţiei de identificare a tipului de piesă sunt obţinute într-o etapă premergătoare aplicaţiei;
  - Senzorul camerei video este paralel cu masa de lucru (proiecţie normală);
- Date de ieşire:
  - Determinarea tipului pieselor aflate în imaginea achiziţionată de camera video;
  - Clasificarea pieselor detectate pe baza unor descriptori de formă ce sunt asociaţi şabloanelor învăţate.

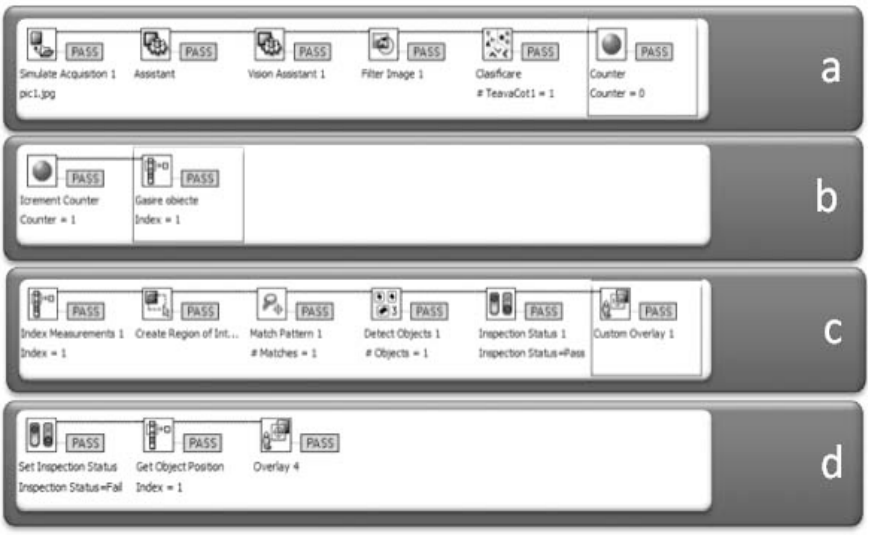


Figura 5. 3: Cele mai reprezentative stări cu operaţiile aferente.

Etapele pe care le presupune această aplicaţie sunt date de rutinele corespunzătoare fiecărei stări.

Etapa 1. Achiziţia şi procesarea imaginii (figura 5.3a). Această etapă presupune următoarele operaţii:

- Achiziţia imaginii;
- Preprocesarea de către camera video în vederea îmbunătăţirii calităţii;
- Procesarea imaginii pentru detectarea obiectelor prezente;

Camera video NI 1741 achiziţionează imagini monocrome ale obiectelor la rezoluţii de 640x480 pix(VGA) şi realizează operaţii de preprocesare a acestora înainte de a fi transmise şi stocate pe calculator. Operaţiile de preprocesare presupun aplicarea unor filtre de îmbunătăţire a semnalului video şi de eliminare a

zgomotului. Acest lucru este posibil datorita faptului că videocamera este prevăzută cu un procesor PowerPC ce funcţionează la o frecvenţă de 533 MHz.

Operaţiile de procesare ce au fost aplicate imaginilor prelevate sunt operaţii de îmbunătăţire a contrastului, de modificare a luminozităţii, de binarizare şi operaţii morfologice (înlăturarea obiectelor mici, închiderea conturului obiectelor, umplerea golurilor).

Înlăturarea obiectelor mici din imaginea binarizată se face prin aplicarea unei operaţii de erodare a obiectelor ce presupune o micşorare a acestora, urmată de o operaţie de dilatare a obiectelor. În urma acestei operaţii morfologice, obiectele ce sunt mai mici decât un prag stabilit (o valoare în pixeli) dispar la erodare, iar cele ce sunt mai mari, revin la starea lor iniţială (valoarea în pixeli pe care o aveau înaintea aplicării erodării). Astfel sunt eliminate din imagine obiectele mici ce nu prezintă interes din punct de vedere al aplicaţiei iar obiectele vizate nu se modifică.

Închiderea morfologică a unui contur este o operaţie ce presupune o dilatare a conturului urmată de o erodare a acestuia. Prin dilatare, un contur deschis, întrerupt, poate să devină închis, astfel încât pixeli de pe conturul imaginar de valoare 0 devin 1, iar în urma erodării aceşti pixeli nu îşi mai modifică valoarea. Umplerea golurilor constă în modificarea valorii pixelilor (0 devine 1) ce se află în interiorul unui contur închis.

De asemenea asupra acestor imagini sunt aplicate şi operaţii de transformare logaritmică, de egalizare de histograma şi de inversare a valori pixelilor.

Egalizarea de histogramă este operaţia de uniformizare a acesteia prin realizarea redistribuţiei de intensitate. Practic domeniul de intensitatea din imagine se extinde, în urma acestei operaţii.

Inversarea valorii pixelilor constă în modificarea acestora, astfel încât pixelii ce au valoarea 1 vor deveni 0 iar cei cu valoarea 0 vor avea valoarea 1. Se realizează practic un negativ al imaginii originale.

Operaţia de transformare logaritmică aplicată asupra imaginii îmbunătăţeşte luminozitatea şi contrastul regiunilor întunecate.

Etapa 2. Clasificarea obiectelor se face în doua faze, respectiv una de instruire şi una de repartizare pe clase.

- În faza de instruire sunt învăţate trei şabloane pentru a putea crea trei clase de obiecte, care sunt utilizate în următoarele fază pentru a identifica obiectele necunoscute din imagini.
- În faza de clasificare, obiectele din imagini sunt asociate pe baza descriptorilor de formă (rectangularitate, lungimea conturului obiectului, aria obiectului) cu una din clase.

Etapa 3. Figura 5.3c, prezintă operaţiile necesare identificării din imaginile achiziţionate a obiectelor ce fac parte din clasa unu. Aceste operaţii sunt:

- de creare a unei regiuni de interes în imagine;
- de detectare a trăsăturilor specifice clasei;
- de actualizare a stării de inspecţie.

Regiunea de interes se creează cu scopul de a simplifica operaţiile ulterioare şi de a diminua timpul de procesare al algoritmului prin faptului că este prelucrată doar o parte din imagine.

Detectarea trăsăturilor specifice clasei constă în extragerea descriptorilor de formă a obiectelor de intensitate omogenă din imagine. Pe baza acestor trăsături, fiecare obiect prezent în imagine, este comparat cu şabloanele pre-învăţate ce sunt asociate claselor de obiecte.

În urma operaţiei de detectare şi de identificare a obiectelor starea de inspecţie se modifică.

Operaţiile specifice identificării obiectelor din clasa 1, se repetă cu mici modificări specifice celorlalte clase pentru identificarea obiectelor rămase.

Etapa 4. Ultima stare, ilustrată în figura 5.3d, asociază obiectele detectate în imagine, ce nu au fost clasificate, dacă acestea există, cu clasa de obiecte necunoscute. Procesul de identificare este complet în momentul în care toate obiectele detectate în imagine au fost catalogate (figura 5.4).

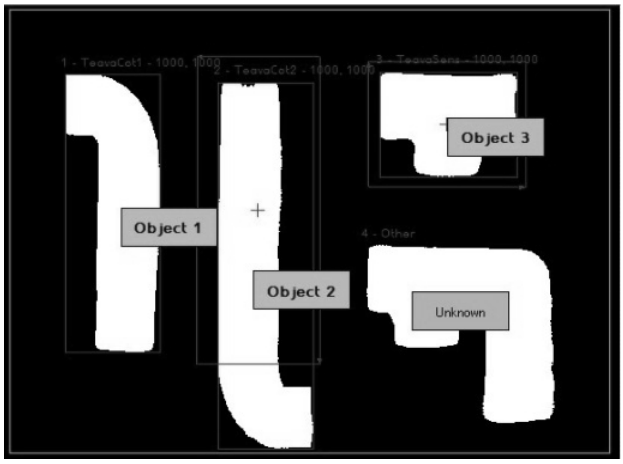


Figura 5. 4: Identificarea obiectelor

Rezultatele obţinute de acest algoritm, în urma rulării pe mai multe imagini, în care tipul, numărul şi poziţia obiectele se modifică, sunt prezentate în figura 5.5.

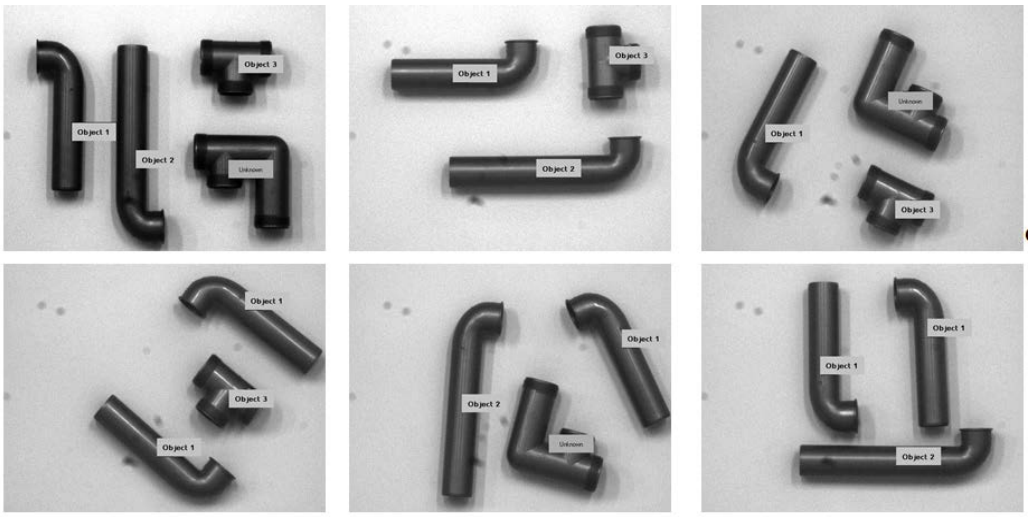


Figura 5. 5: Rezultatele procesului de identificare

5.2. Aplicaţie de vedere robotizată

Scopul aplicaţiei: Identificarea unui obiect 2D, de tip placă, în timp real pe baza unui algoritm de vedere computerizată, pentru a putea fi manipulat de către un braţ robotic (Pop, 2011a).

5.2.2. Generalităţi

Dintre aplicațiile de asamblare cele mai uzuale bazate pe roboți industriali și sisteme de vedere artificială putem distinge ca şi categorie de operații cele de "pick and place".

Maşinile utilizate pentru executarea operațiile de „pick-and-place" sunt folosite într-o varietate largă de aplicaţii în întreaga industrie a produselor de consum. Viteza şi precizia cu care trebuie să îndeplinească sarcinile care le revin în mod direct afectează productivitatea, şi în cele din urmă, linia de fabricaţie. Aceste mașini menționate anterior au rolul de a ridica obiectele și de a le muta dintr-o locație în alta. Exemple de astfel de maşini pot fi: sistemele portabile de manipulare a materialelor, manipulatoarele industriale, roboţi cu funcţia de manipulare, sistemele automate de manipulare a materialelor, sistemele de manipulare a materialelor specifice fabricilor şi maşinile de "pick and place".

Viteza şi precizia sunt printre principalele obiective ce trebuie avute în vedere atunci când se proiectează un sistem de tip „pick-and place". Cu cât este mai strict şi mai simplu sistemul de control al mişcării cu atât acesta poate rula mai rapid şi mai eficient.

Roboții cartezieni sunt folosiți în operații mai simple de „pick and place". Câteva exemple de astfel de operaţii sunt:

- Asamblare de circuite electronice;
- Plasare de obiecte pe o bandă transportoare sau pe un cărucior;
- Ambalarea diferitelor produse.

Aplicaţia prezentată în această secţiune presupune detectarea unui obiect de tip placă, prezent în spaţiul de lucru al robotului şi determinarea poziţiei sale faţă de un sistem de referinţă, în vederea conducerii unui robot (pe baza unei traiectorii generate). Standul experimental este prezentată în figura 5.6 şi conţine:

- Un sistem robotizat compus din:
  - Robot cartezian;
  - Controler robot;
  - Cutie conexiuni
  - Calculator industrial de comandă şi control.
- Un sistem de vedere artificială ce are în componenţă:
  - O cameră industrială inteligentă de tip „smart camera" de la National Instruments;
  - O placă de achiziţie;
  - Calculator pentru dezvoltarea şi rularea algoritmului de vedere computerizată;
  - O sursă de lumină;
  - Programul de bază LabView (mediul de lucru în care s-a dezvoltat aplicaţia).
- Piesă de tip „placă" la care o dimensiune este constantă

De asemenea s-a mai utilizat şi un şablon de calibrare de tip „grilă de puncte" (dot grid) pentru calibrarea camerei video.

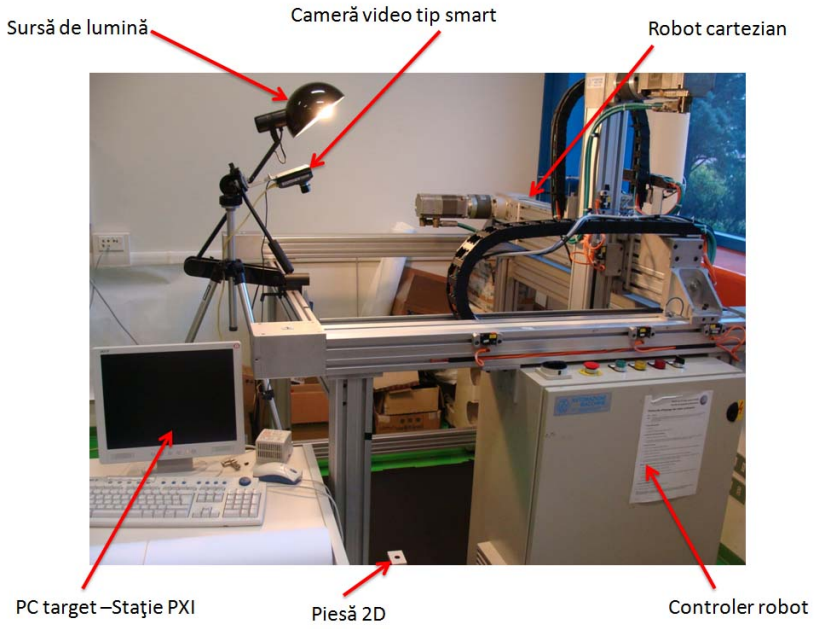


Figura 5. 6: Stand experimental

Robotul cartezian utilizat este prevăzut cu trei axe de translaţie corespunzătoare a trei grade de libertate şi dotat cu un efector final atașat axei verticale care îi conferă încă trei grade de libertate. Această structură oferă caracteristici excelente de rigiditate mecanică.

Sistemul de comandă al robotului prezintă o configurație de tip „xPc Target" și este alcătuit din:

- Stația PXI NI 1042Q (PC Target) de la National Instruments echivalentă a unui calculator industrial caracterizat printr-un procesor de mare viteză necesar operațiilor de testare și control „real-time";
- Un calculator pentru prelucrarea datelor primite de la PXI (PC Host);
- Rețele de comunicare bazate pe protocolul de internet TCP/IP.

Camera inteligentă (smart camera) reprezintă un sistem de vedere de sine stătător având încorporat un senzor de imagine într-o carcasă robustă de tipul unei camere industriale. Acesta conţine toate interfeţele necesare de comunicare, de exemplu, Ethernet, precum şi linii de tensiune de 24V- I/O (intrare/ieşire) pentru conectarea la un PLC, la actuatoare, relee sau supape pneumatice. Nu depăşeşte dimensiunile unei camere normale de inspecţie şi supraveghere.

Aceste camere au un procesor dedicat şi sunt potrivite în special pentru aplicaţii în cazul în care mai multe camere trebuie să funcţioneze independent şi, adesea, asincron, sau atunci când este necesar un sistem de vedere distribuită (spre exemplu pentru inspecţie sau în cazul în care sunt necesare mai multe puncte de supraveghere de-a lungul unei linii de producţie sau într-un sistem de asamblare).

5.2.2. Aplicaţia propriu zisă

Etapele premergătoare realizării acestei aplicaţii sunt calibrarea camerei video şi învăţarea unui şablon corespunzător piesei utilizate.

Camera video NI 1742 este fixată într-o poziţie paralelă cu planul XY al spaţiului de lucru al robotului cartezian. Calibrarea camerei se face faţă de un sistem de referinţă fix, denumit sistemul de referinţă global a cărui situare se cunoaşte faţă de sistemul de referinţa ataşat robotului.

Calibrarea camerei se realizează cu ajutorul modulului de calibrare existent în LabView şi cu ajutorul şablonului de calibrare amintit anterior. Paşi parcurşi pentru efectuarea acestei operaţii sunt:

- Plasarea şablonului de calibrare în câmpul vizual al camerei video;
- Achiziţia unei imagini cu şablonul de calibrare;
- Rularea modulului de calibrare ce presupune;
    - Prelucrarea imaginii în vederea detectării punctelor de calibrare ale şablonului;
    - Ataşarea unui sistem de referinţă şablonului care să corespundă cu sistemul de referinţă global;
    - Eliminarea distorsiunilor introduse de camera video;
    - Stabilirea unei corespondenţe între distanţa în pixeli dintre două puncte şi distanţa reală în milimetri;
    - Alegerea unui algoritm de calibrare.

Informaţiile obţinute în urma operaţiei de calibrare a camerei sunt asociate imaginii utilizate şi sunt stocate în memorie pentru ca ulterior să poată fi aplicate imaginilor ce conţin piesa 2D.

De asemenea sunt stocate şi informaţiile cu privire la descriptorii de formă asociate şablonului piesei 2D ce urmează a fi identificată şi manipulate de robot. Şablonul se realizează cu ajutorul algoritmului „IMAQ Learn Pattern" existent în LabView, ce presupune utilizarea metodei SHIFT pentru extragerea trăsăturilor locale ce nu se modifică în imagine în urma aplicării unor operaţii geometrice precum rotire sau translatare.

Figura 5. 7: Şablon piesă 2D

Algoritmul de vedere computerizată, dezvoltat de autor în LabView, pentru detectarea, recunoaşterea şi determinarea situării obiectului faţă de sistemul global, este un algoritm grafic ce se regăseşte la anexa 5.

Paşii ce sunt necesari a fi parcurşi pentru realizarea aplicaţiei sunt următorii:

Pasul 1. Plasarea piesei 2D în spaţiul de lucru al robotului într-o poziţie necunoscută;

Pasul 2. Achiziţia unei imagini cu piesa ce urmează a fi manipulată;

Pasul 3. Preprocesarea imaginii de către camera video în vederea îmbunătăţirii acesteia prin eliminarea zgomotului;

Pasul 4. Încărcarea informaţiilor obţinute în urma calibrării camerei video;

Pasul 5. Prelucrarea imaginii în vederea identificării obiectului în cauză.

Aplicarea unor operaţii de filtrare a imaginii (eliminarea artefactelor), de egalizare de histogramă (ce permite îmbunătăţirea contrastului), de binarizare pe baza unui prag prestabilit, de detectare de muchii şi de inversare a valori pixelilor.

Pasul 6. Încărcarea imaginii cu şablonul corespunzător piesei şi implicit a informaţiilor cu privire la descriptorii de formă.

Pasul 7. Identificarea obiectului pe baza şablonului şi determinarea centrului de greutate al acestuia;

Operaţia de identificare a obiectului se realizează utilizând funcţia „Geometric Matching" (potrivire geometrică). Această funcţie a fost dezvoltată pentru a localiza într-o imagine, obiecte care să se potrivească cu un model predefinit. Funcţia „Geometric Matching" permite localizarea obiectelor corespunzătoare şabloanelor învăţate pe baza unor descriptori de formă. Această funcţie poate identifica obiecte, chiar şi în cazul în care avem variaţii de iluminare, zgomot, şi transformări geometrice de tip deplasare, rotaţie sau scalare.

Pasul 8. Determinarea poziţiei piesei (coordonatele acesteia) utilizând informaţia de calibrare a camerei;

Pasul 9. Pe baza coordonatelor obţinute se generează traiectoria necesară pentru conducerea robotului. Generarea traiectoriei se realizează cu ajutorul unui algoritm, dezvoltat de autor, ce se regăseşte la anexa 6.

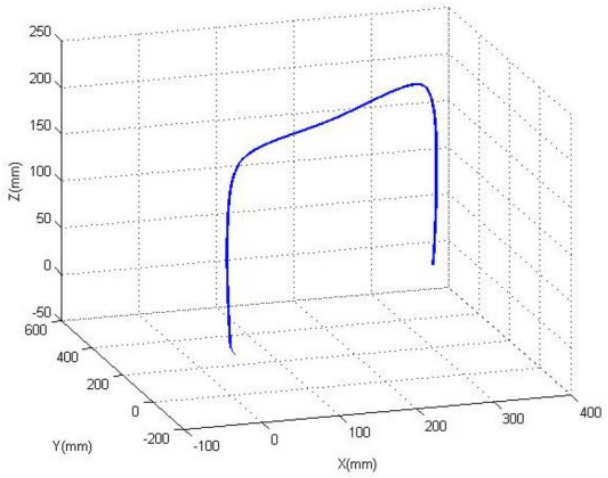


Figura 5. 8: Reprezentarea grafică a traiectoriei generate

Conducerea robotului se face prin intermediul programului LabView Real Time Module de la National Instruments ce rulează pe staţia PXI. Acest program presupune rularea unor aplicaţii specifice.

Dezvoltarea unei astfel de aplicaţii se realizează în mod asemănător celor din LabView (tot pe baza de cod grafic), cu diferenţa că acestea necesită câteva funcţii adiţionale particulare sistemelor real-time:

- Funcţii „Watchdog" (timer) pentru a reporni în mod automat unele componente hardware în cazul în care programul se opreşte din funcţionare;
- Funcţii de a comunica datele în mod determinist între părţile componente ale unui program în timp real;

- Funcţii utilitare pentru a configura o încărcare echilibrată pe sisteme cu mai multe nuclee de procesare;
- Funcţii de sincronizare pentru a controla cu precizie execuţia de bucle în programe de timp real-time.

Pentru conducerea robotului cartezian, utilizat în această aplicaţie, a fost dezvoltat în cadrul colectivului de cercetare de la Universitatea din Udine un program de tip real-time. Acesta implică generarea unui fişier text (cu extensia .txt) pe baza traiectoriei obţinute, cu informaţiile necesare encoderelor.

Paşii pe care îi presupune acest program sunt următorii:

Pasul 1. Datele despre traiectorie sunt încărcate de pe un fişier în format foaie de calcul (coloanele sunt separate printr-un "tab", rândurile se disting printr-un semn EOL);

Pasul 2. Se construieşte un control în buclă, bazat pe PID (Proportional-Derivative-Integral);

Pasul 3. Se scriu datele encoder-ului într-un fişier în format foaie de calcul;

Pasul 4. Se creează o interfaţă grafică.

Programul general este compus din 3 tipuri de bucle:

- Prima buclă citeşte datele din fişierul ce conţine informaţiile despre traiectorie;
- A doua buclă conţine sisteme de tip „DAQ- assistant” şi funcţii de control;
- A treia buclă scrie datele de pe un encoder pe un fişier de tip text.

Bucla 1, poate fi împărţită în 5 faze distincte.

În faza 1, se creează un document nou. Deoarece scopul este de a citi dintr-un fişier de tip „spreadsheet” acest bloc va deschide un fişier. txt. Acest fişier. txt trebuie să fie salvat pe sistemul Real Time-target şi este transferat printr-un protocol de tip ftp.

Transferul acestui fişier între PC-ul gazdă (host) şi ţintă (target) se poate face prin orice program ftp. În cazul de faţă s-a utilizat NI MAX (National Instruments Measurement and Automation Explorer), care este folosită pentru că oferă o privire de ansamblu a terminalelor instalate pe fiecare sistem de tip host şi de asemenea conţine un instrument ftp ce este util de folosit pentru încărcarea automată a tuturor setările de tip ftp.

A doua fază constă în deschiderea unui fişier de tip queue („coadă de aşteptare”). Deoarece buclele au viteze diferite şi fişierele au nevoie de date la un anumit moment, acestea trebuie să fie puse într-un şir de aşteptare.

Transferul de date prin intermediul unui fişier de tip „coadă de aşteptare” de obicei este compus din 4 paşi.

- Se iniţializează „coada de aşteptare” (faza 3). Se selectează un format de fişier şi un nume;
- Se pun datele în coadă de aşteptare (faza 4);
- Se extrag datele din „coada de aşteptare” (faza 4);
- Se eliberează coada de aşteptare (faza 5). Acest lucru se realizează de obicei atunci când coada de aşteptare nu mai conţine niciun fişier de date.

Buclă 2 este numită "buclă temporizată”, se execută la o anumită frecvenţă şi împărţită într-un anumit interval de timp. De asemene această buclă conţine trei algoritmi similari, pentru toate cele trei axe ale robotului X,Y şi Z.

Bucla 3 este creată în scopul de a scrie datele de pe encoder, care este exportat din bucla temporizată de o coadă de aşteptare, într-un fişier text. Funcţionează în mod analog cu prima buclă doar că în sens invers – scrie datele în loc să le citească.

La final este creat sau înlocuit un nou fişier de tip text de pe sistemul Real Time Target, care va servi ca spaţiu de stocare pentru datele encoder-ului.

În figura 5.9 este prezentată interfaţa utilizatorului pentru realizarea controlerului robotului.

Poziţiile 2, 3 şi 4 din figura 5.9 reprezintă intrările pentru interfaţa utilizator care constau în:

- Amplificatoarele de semnal pentru cele trei axe – numărul 2;
- Căile de fişier pentru datele de intrare şi ieşire – numărul 3;
- Raportul de transmisie - numărul 4.

Poziţiile 1,5 şi 6 din figura 5.9 reprezintă ieşirile acestei interfeţe:

- Grafice care reprezintă datele encoder (1);
- Indicaţie numerică a distanţei în mm (5);
- Valorile numerice ale datelor encoder-ului (6).

Butonul de stop opreşte execuţia traiectoriei în orice moment.

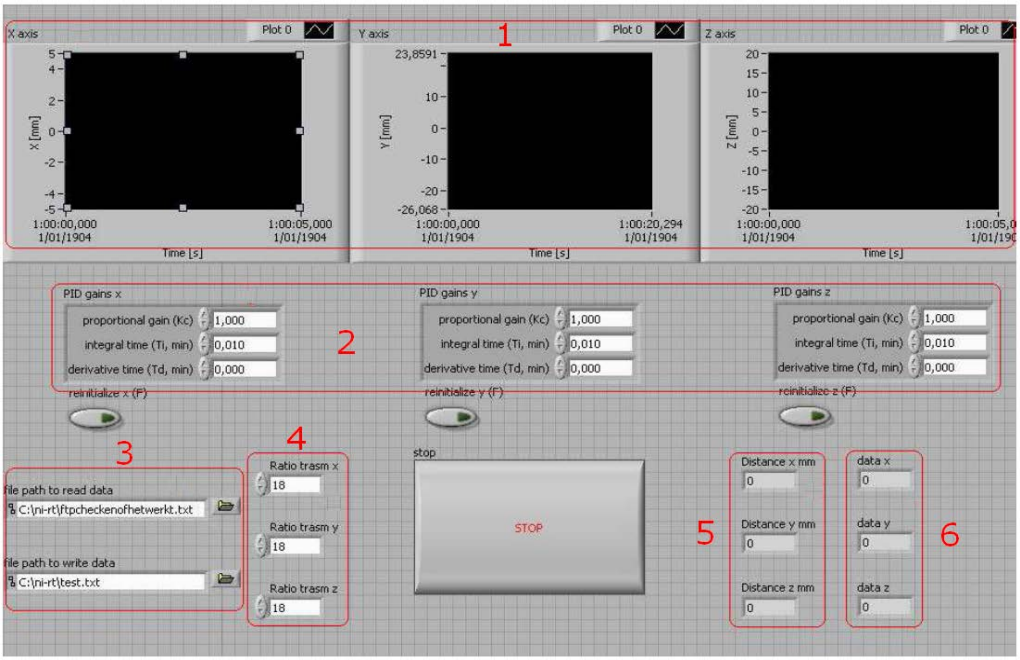


Figura 5. 9: Interfaţă grafică a aplicaţiei pentru conducerea robotului

5.3. Concluzii

În acest capitol, a fost prezentat un sistem de conducere a unui robot pe baza informaţiilor extrase din imagini şi o aplicaţie de recunoaştere şi clasificare de obiecte. S-a putut observa că modalitatea de conducere a robotului este funcţională şi precisă, iar performanţele aplicaţiei de recunoaştere sunt corespunzătoare, fiind verificate cu succes pe mai multe imagini. S-a demonstrat că, pentru un număr limitat de obiecte, a căror formă nu este foarte complexă, în anumite condiţii specificate, sistemul funcţionează.

6. APLICAŢIE DE DETECTARE ŞI RECUNOAŞTERE A UNOR OBIECTE ÎN VEDEREA MANIPULĂRII ROBOTIZATE

În cadrul acestui capitol, este prezentată o aplicaţie de vedere robotizată, ce a fost proiectată şi implementată în mediul de lucru Matlab, pentru recunoaşterea unor obiecte pe bază de înălţime, cu scopul de sortare, manipulare stabilă şi stocare corespunzătoare a acestora într-un depozit de piese.

6.1. Introducere

Scopul aplicaţiei: Detectarea obiectelor ce sosesc pe căruciorul unui conveior de transfer şi determinarea unei caracteristici geometrice specifice precum înălţimea acestora. Pe baza înălţimii, se realizează determinarea tipului de obiect pentru prestarea operaţilor ulterioare de sortare şi depozitare, dar şi obţinerea unei prinderi fixe, a obiectului, între bacurile dispozitivului de prehensare, astfel încât operaţia de manipulare să se efectueze într-un mod stabil şi controlat.

Figura 6. 1: Piese colorate.

Pentru realizarea acestei aplicaţii, au fost utilizate piese colorate (figura 6.1) precum şi echipamentele şi dispozitivele din laboratorul CIM (Computer Integrated Manufacturing) ce se regăseşte în cadrul departamentului de Mecatronică. Aceste echipamente sunt (figura 6.2):

- Sistemul de vedere artificial compus din:
  - Camera de luat vederi (cameră video CCD);
  - Placă de achiziţie;
  - Calculator destinat prelucrării de imagini;
- PLC-ul (Programmable Logic Controler);

- Conveior de transfer al pieselor între staţii prevăzut cu patru cărucioare şi trei posturi de aşteptare;
- Staţia de asamblare ce cuprinde un robot de tip SCARA care realizează operaţia de manipulare a pieselor.
- Sistemul de comandă format dintr-un calculator central şi un controler central care comandă controlere de staţii. Comunicarea se realizează prin reţea serială RS 232.

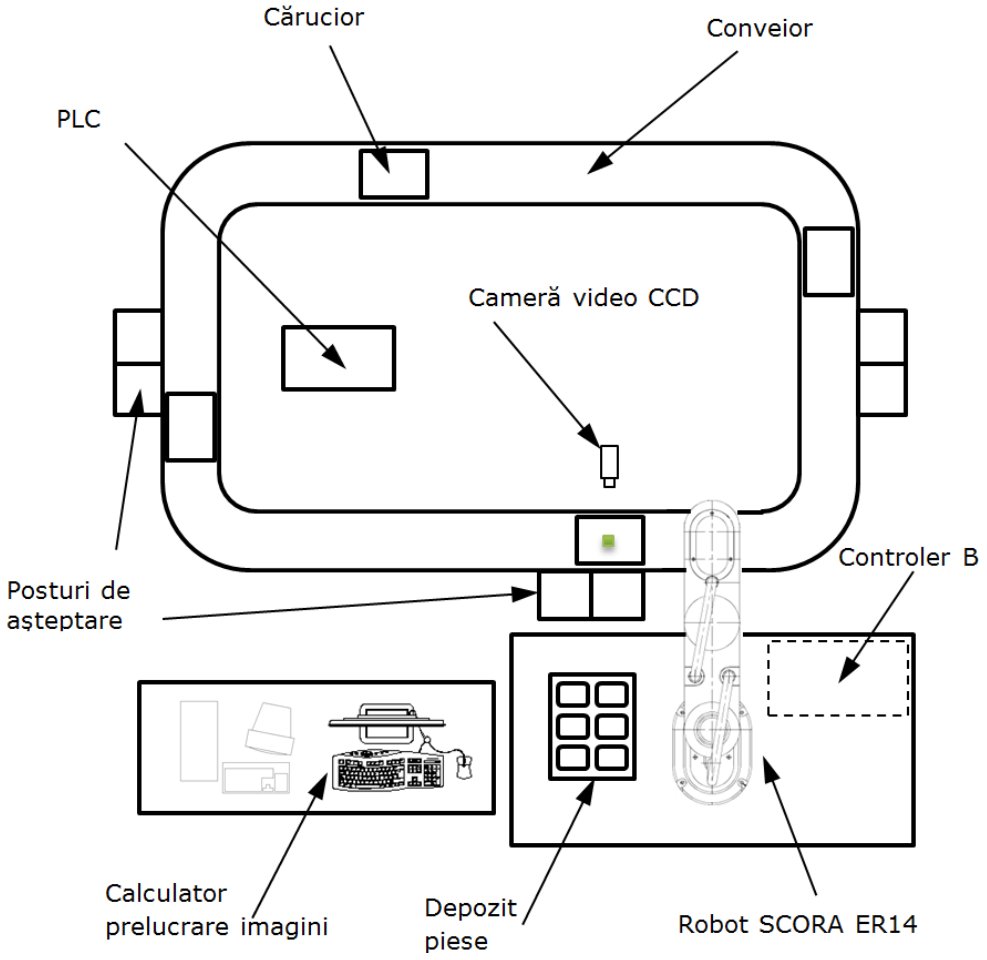


Figura 6. 2: Schema de principiu a echipamentelor şi dispozitivelor utilizate, aflate în dotarea laboratorului CIM.

Etapele premergătoare realizării aplicaţiei de detectare, recunoaştere şi manipulare a obiectelor colorate sunt:

- Calibrarea robotului SCORA ER 14;
- Calibrarea camerei video;
- Determinarea modalităţii de conducere a robotului;
- Alegerea metodei de detectare a obiectelor;
- Selectarea modalităţii de recunoaştere a obiectelor

6.2. Calibrarea robotul SCORA ER-14

6.2.1. Aspecte generale

Una dintre cele mai importante componente ale standului experimental o reprezintă robotul SCORA ER 14 de la firma ESHED ROBOTEC. Acesta este un robot de tip SCARA (Selective Compliant Assembly Robot Arm sau Selective Compliant Articulated Robot Arm) prevăzut cu patru axe robuste, trei cuple de rotaţie şi una de translaţie.

Robotul scara, este un robot cu structură simplă, foarte frecvent întâlnit în operaţiile de asamblare, manipulare şi ambalare a produselor cu sarcină utilă mică şi dimensiuni de gabarit reduse. Se utilizează des în operaţiile industriale deoarece realizează acest tip de operaţii cu o acurateţe impresionantă la viteze ridicate.

SCORA ER-14 (figura 6.3 a), este de tip selectiv deoarece se poate mişca cu uşurinţă în plan orizontal dar nu şi în plan vertical, fapt datorat structurii sale geometrice. Din punct de vedere constructiv, se compune din două elemente rigide conectate între ele prin cuple cinematice de rotaţie ce permit efectuarea mişcării în planul XY şi un al treilea element conectat printr-o cuplă de translaţie ce realizează mişcarea pe verticală (figura 6.3 b). Efectorul final est prevăzut cu o cuplă de rotaţie ce permite dispozitivului de prehensiune (gripper-ului) rotirea în jurul propriei axe.

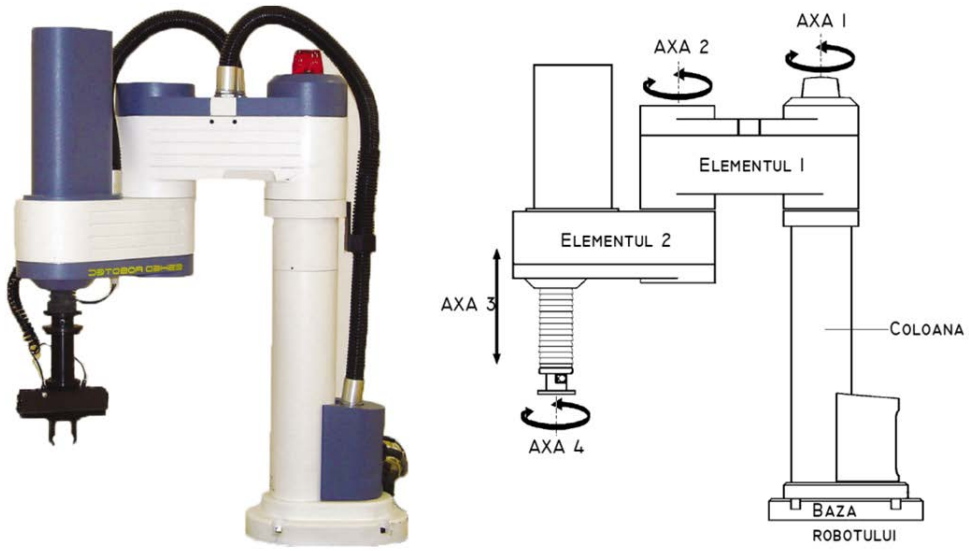


Figura 6. 3: a) Robotul SCORA ER 14; b) Elementele componente.

6.2.2. Modelarea geometrică

În vederea obţinerii unei precizii de situare (poziţionare şi orientare) ridicată, ce este indispensabilă în orice aplicaţie industrială, este necesar ca robotul SCORA ER 14 să fie calibrat. Prin calibrarea robotului, se înţelege procesul prin care

se determină parametri cinematici şi dinamici ai acestuia. Prima etapă a acestei operaţii o constituie modelarea geometrică.

Modelarea geometrică a robotului SCORA ER 14 implică, ca la orice robot, reprezentarea acestuia sub forma unei configuraţii de corpuri rigide ce sunt legate între ele prin cuple de rotaţie şi translaţie. Modelarea se realizează pentru a determina de fapt una din condiţiile funcţionale ale robotului, respectiv situarea efectorului final în funcţie de situările relative ale elementelor sale componente.

Această condiţie se stabileşte pe baza unor relaţii matematice ce exprimă legătura dintre coordonatele operaţionale şi coordonatele cuplelor cinematice.

Din punct de vedere mecanic, robotul este un lanţ cinematic deschis, iar pentru a determina situarea (poziţia şi orientarea) efectorului final în raport cu un sistem de referinţă, este necesar să se realizeze analiza cinematică. Această analiză presupune o transformare dintr-un sistem de coordonate în altul (cel operaţional şi cel al cuplelor cinematice), respectiv determinarea matricei de trecere dintr-un sistem de referinţă în alt sistem de referinţă.

Rezolvarea analizei cinematice a robotului SCORA ER 14 constă în definirea şi soluţionarea problemei cinematice directe sau a problemei cinematice inverse.

Pentru rezolvarea celor două probleme este necesar să se cunoască parametrii dimensionali (figura 6.4) ai elementelor robotului şi rapoartele de transmitere.

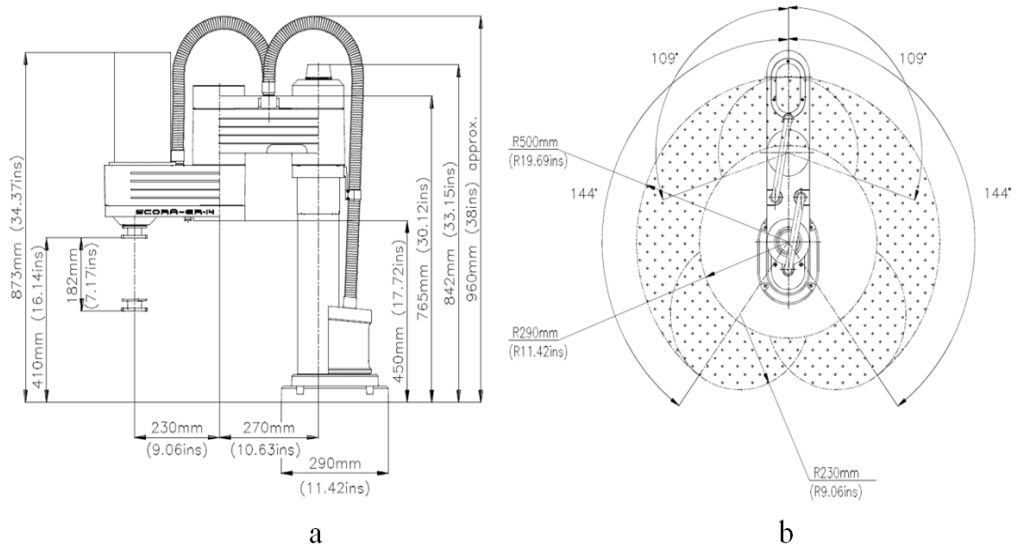


Figura 6. 4: a) Dimensiunile de gabarit şi dimensiunile elementelor componente; b) Spaţiul de lucru.

Problema cinematică directă presupune un ansamblu de relaţii ce permit definirea poziţiei relative a elementelor lanţului cinematic al robotului şi care implică determinarea situării (poziţia şi orientarea) sistemului de coordonate ataşat efectorului final în funcţie de coordonatele articulare (interne) faţă de sistemul de coordonate ataşat bazei robotului. Practic se realizează o transformare a coordonatelor articulare în coordonate operaţionale.

Soluţionarea problemei cinematice directe se poate face utilizând convenţia Hartenberg-Denavit care costă în (Hartenberg & Denavit, 1964):

- Identificarea şi numerotarea cuplelor cinematice;
- Numerotarea elementelor;
- Alegerea sistemelor de referinţă ce se ataşează fiecărui element;
- Stabilirea parametrilor geometrici specifici elementelor şi cuplelor cinematice;
- Definirea matricelor $^{i-1}A_i$ care descriu situarea relativă a elementului (i) în raport cu sistemul de referinţă ataşat elementului (i-1);
- Calcularea matricei $^{0}T_4$ ce redă situarea elementului 4 (efectorul final) în raport cu sistemul de referinţă ataşat bazei robotului.

Parametrii geometrici ce caracterizează elementele robotului SCORA ER 14 sunt:

- Lungimea elementelor (a1 şi a2): se referă la distanţa dintre cuplele cinematice ce se găsesc la capătul acestora; se măsoară de-a lungul perpendicularei comune a axelor cuplelor cinematice.
- Răsucirea elementului (α): reprezintă unghiul format de axele cuplelor de la capătul elementelor.

Parametrii geometrici ce caracterizează cuplele cinematice ale robotului SCORA ER 14 sunt:

- Unghiul dintre două perpendiculare (θ_1, θ_2 şi θ_4): se măsoară în plan perpendicular pe axa cuplei;
- Distanţa d_i (d_1, d_2 şi d_4) măsurată pe axa cuplei (i) între picioarele perpendicularelor comune axelor (i-1) şi (i), respectiv (i) şi (i+1).

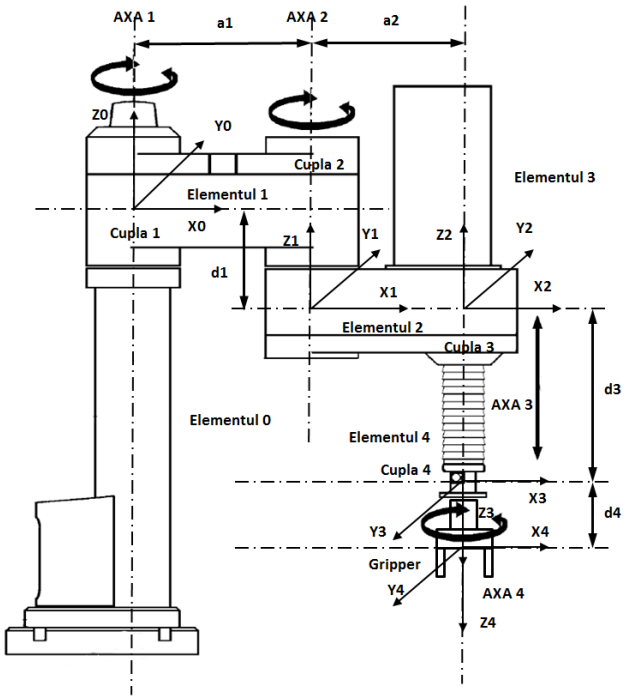


Figura 6. 5: Reprezentarea grafică a cuplelor cinematice, elementelor şi parametrilor geometrici specifici robotului SCORA ER 14.

Sistemul de referinţă ataşat efectorului final este definit de trei versori:

- $\bar{a}$ versorul de apropiere: are direcţia dreptei caracteristice şi defineşte axa O_4z_4 a sistemului ataşat;
- $\bar{o}$ versorul de orientare : are direcţia dreptei auxiliare şi defineşte axa O_4y_4 a sistemului ataşat;
- $\bar{n}$ ($\bar{n}= \bar{o}\times\bar{a}$) versorul norma: precizează baza ortonormată şi defineşte axa O_4x_4 a sistemului ataşat;

Pentru a se soluţiona problema cinematico-poziţională directă a robotului SCORA ER 14 a fost necesar să se determine unghiurile θ_1 şi θ_2 (θ_4 nu s-a calculat). S-a considerat poziţia finală a robotului ca fiind dată de punctul M de coordonate (M_x, M_y)=(-27.758, 287.380) şi s-a reprezentat grafic în vederea stabilirii relaţiilor de calcul.

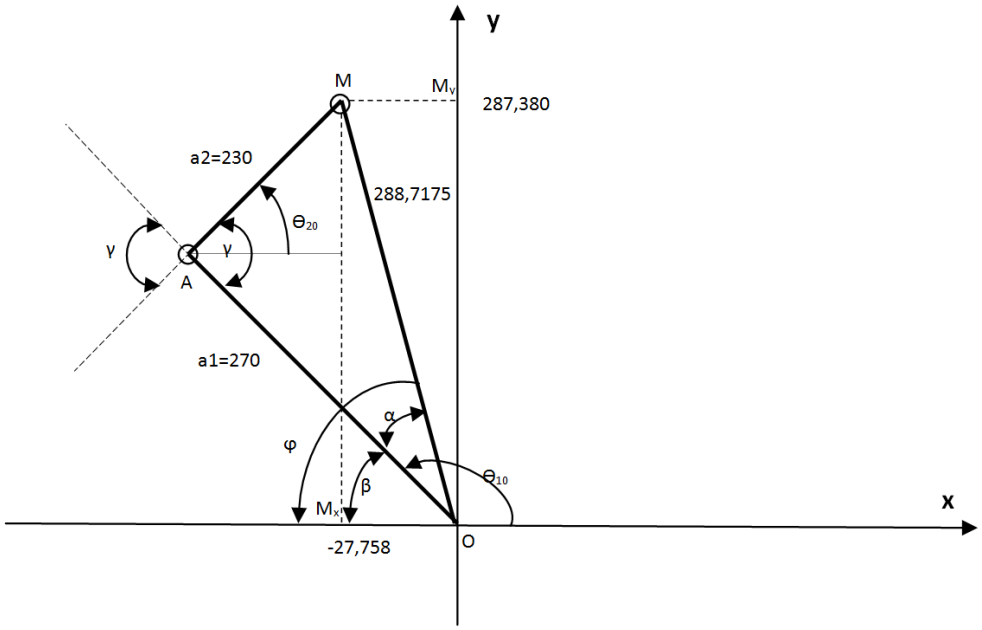


Figura 6. 6: Reprezentare grafică a punctului M.

- S-a determinat OM ca fiind ipotenuza triunghiul dreptunghic OM_xM:

$$M_xM = OM_y$$
$$OM = \sqrt{OM_x^{\,2} + M_xM^2} \tag{6.1}$$

- S-a calculat unghiul φ:

$$\varphi = \tan^{-1}\frac{MM_x}{OM_x} \tag{6.2}$$

- S-a obţinut unghiul γ ca:

$$\gamma = \cos^{-1}\frac{MA^2 + AO^2 - OM^2}{2*MA*AO} \tag{6.3}$$

- S-a determinat unghiul α:

$$\alpha = \cos^{-1}\frac{OA^2 + OM^2 - AM^2}{2*OA*OM} \tag{6.4}$$

- Pe baza unghiurilor φ , α şi γ obținute anterior s-au putut determina unghiurile θ_{10} şi θ_{20} :

$$\theta_{10} = 180^0 - (\varphi - \alpha) \tag{6.5}$$

$$\theta_{20} = (180^0 + \theta_{10} + \gamma) - 360^0 \tag{6.6}$$

- Utilizând numărul de impulsuri, date de traductoarele de poziție, ce corespund coordonatelor punctului M, şi parametrul robotului SCOA ER 14 a cărui valoare este numărul de impulsuri la traductorul de poziţie al axei comandate pentru o rotaţie cu +90º, s-a putut calcula $\Delta\theta_1$ şi $\Delta\theta_2$:

$$\Delta\theta_1 = \frac{imp1 * 90^\circ}{par}$$
$$\Delta\theta_2 = -\frac{imp2 * 90^\circ}{par} \tag{6.7}$$

- Pe baza unghiurile θ_{10} , θ_{20} , $\Delta\theta_1$, $\Delta\theta_2$ şi γ obţinute în paşii anteriori, s-a calculat unghiul θ_1 şi unghiul θ_2 :

$$\theta_1 = \Delta\theta_1 + \theta_{10}$$
$$\theta_2 = \Delta\theta_2 + \theta_{20} + \gamma \tag{6.8}$$

În urma rezultatelor obținute şi pe baza parametrilor iniţial cunoscuţi s-a completat tabelul parametrilor Hartenberg-Denavit.

Cuple cinematice	Unghiul θ [grade]	Lungimea a[mm]	Distanţa d[mm]	Răsucirea α[grade]
1	θ_1	a1 (270)	d1 (125)	0
2	θ_2	a2 (230)	0	0
3	0	0	d3	α3 (180)
4	θ_4	0	d4 (140)	0

Tabelul 6. 1: Parametrii Hartenberg-Denavit.

Utilizând parametri din tabelul de mai sus se determină matricele de trecere de la un sistem de referinţă la altul, respectiv trecerea de la (i+1) la (i). Astfel în cazul robotului SCORA ER 14 se pot construi patru matrici, una pentru fiecare cuplă cinematică.

$$ {}^{0}A_{1} = \begin{bmatrix} \cos\theta_1 & -\sin\theta_1 & 0 & a_1\cos\theta_1 \\ \sin\theta_1 & \cos\theta_1 & 0 & a_1\sin\theta_1 \\ 0 & 0 & 1 & d_1 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{6.9} $$

$$ {}^{1}A_{2} = \begin{bmatrix} \cos\theta_2 & \sin\theta_2 & 0 & a_2\cos\theta_2 \\ \sin\theta_2 & -\cos\theta_2 & 0 & a_2\sin\theta_2 \\ 0 & 0 & -1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{6.10} $$

$$ {}^{2}A_{3} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & d_3 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{6.11} $$

$$ {}^{3}A_{4} = \begin{bmatrix} \cos\theta_4 & -\sin\theta_4 & 0 & 0 \\ \sin\theta_4 & \cos\theta_4 & 0 & 0 \\ 0 & 0 & 1 & d_4 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{6.12} $$

Trecerea de la sistemul de referinţă ataşat efectorului final la cel ataşat bazei robotului se face cu matricea de transformare omogenă, conform relaţiei de recurenţă (notăm funcţia sin cu s şi cos cu c):

$$ {}^{0}T_{4} = {}^{0}A_{1} \cdot {}^{1}A_{2} \cdot {}^{2}A_{3} \cdot {}^{3}A_{4} $$

$$ {}^{0}T_{4} = \begin{bmatrix} s\theta_4\, s(\theta_1+\theta_2) + c\theta_4\, c(\theta_1+\theta_2) & s\theta_4\, c(\theta_1+\theta_2) + c\theta_4\, s(\theta_1+\theta_2) & 0 & a_2\, c(\theta_1+\theta_2) + a_1\, c\theta_1 \\ s\theta_4\, c(\theta_1+\theta_2) + c\theta_4\, s(\theta_1+\theta_2) & -s\theta_4\, s(\theta_1+\theta_2) + c\theta_4\, c(\theta_1+\theta_2) & 0 & a_2\, s(\theta_1+\theta_2) + a_1\, s\theta_1 \\ 0 & 0 & -1 & -d_4 - d_3 + d_1 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{6.13} $$

6.2.3. Modelul matematic al determinării matricei de trecere

Calibrarea robotului SCORA ER 14, pentru aplicaţia de faţă, implică realizarea unei corelaţii, reciproc valabile, între coordonatele spaţiului în care se găsesc piesele ce urmează a fi manipulate de robot şi coordonatele robotului. Această legătură reciprocă se concretizează prin determinarea unui model matematic al matricei de transformare omogene (Grigorescu, 2012).

Pentru soluţionarea modelului matematic al matricei de trecere se consideră două sisteme de referinţă carteziene, fiecare format din trei plane de referinţă reciproc ortogonale, respectiv doua sisteme tri-ortogonale:

- Un sistem $O_R X_R Y_R Z_R$ ataşat robotului SCORA ER 14;
- Un sistem $O_P X_P Y_P Z_P$ ataşat spaţiului de lucru (spaţiul în care se găsește piesa).

Pe baza condiţiilor amintite anterior, se poate afirma că este necesar şi suficient pentru definirea sistemul de coordonate ataşat spaţiului de lucru (spațiul

piesei) să se stabilească trei puncte a căror poziţie să fie cunoscută faţă de sistemul ataşat bazei robotului.

Aceste puncte cunoscute ce se aleg sunt (figura 6. 7):

- Originea sistemului de referinţă ataşat spaţiului piesei (O_P);
- Un punct care defineşte sensul pozitiv al axei X al noului sistem (în acest caz punctul A ce se regăseşte pe axa O_PX_P);
- Un punct ales la întâmplare în plan, a cărui coordonate faţă de sistemul piesei sunt pozitive.

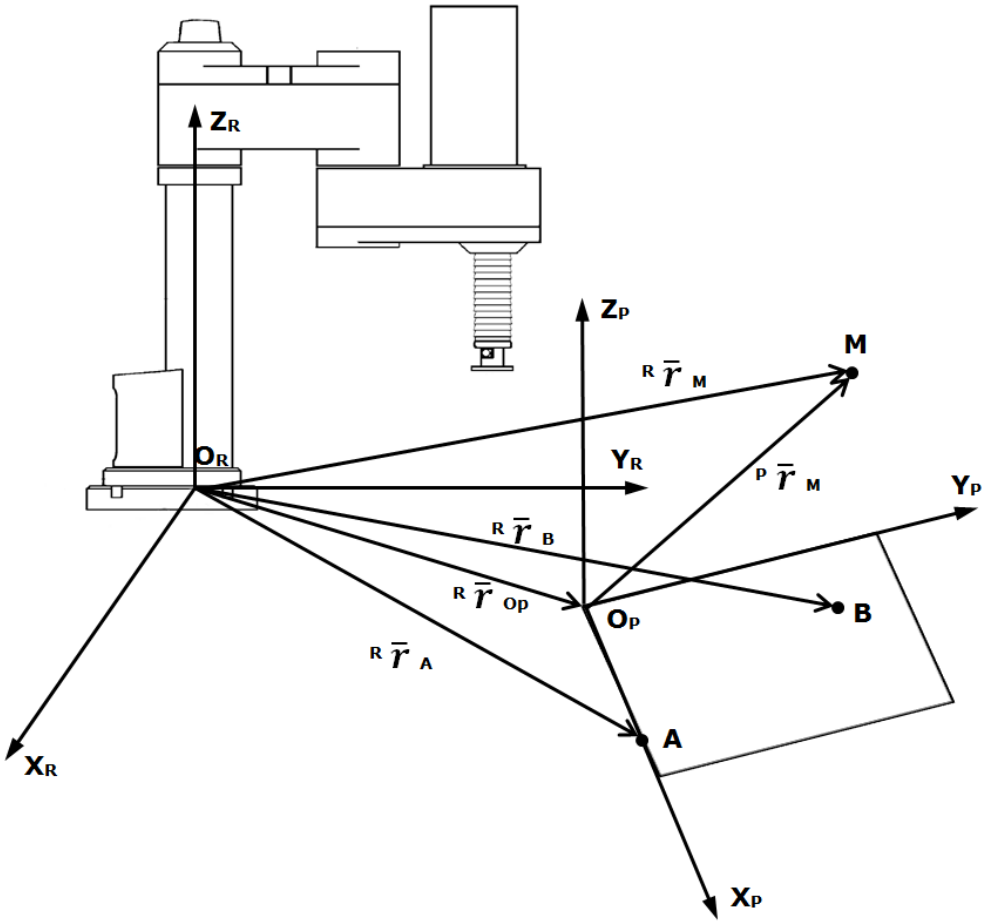


Figura 6. 7: Sistemele de referinţă şi punctele necesare transformării de coordonate

Cele trei puncte necesare se determină prin deplasarea gripper-ului robotului, în care se găseşte prins un ştift, în planul piesă (planul orizontal din spaţiul de lucru) de care este ataşat sistemul de referinţă „P”. Se poziţionează cu foarte mare acurateţe ştiftul în cele trei puncte menţionate anterior, astfel încât să se cunoască precis coordonatele acestora faţă de sistemul „P”. Această operaţie s-a realizat prin utilizarea unei grile de calibrare (vezi figura 6.8). Determinarea coordonatelor punctelor faţă de sistemul „R” s-a realizat prin apelarea în programul controlerului robotului a comenzii „LISTPV POSITION”. Comanda permite afişarea

impulsurilor traductoarelor de poziţie ale motoarelor robotului şi coordonatele în cartezian faţă de sistemul ataşat bazei robotului.

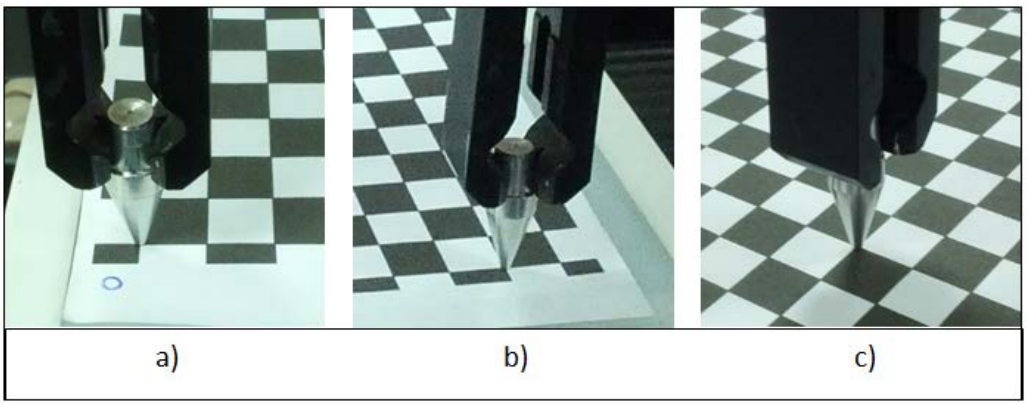


Figura 6. 8: Poziţionarea ştiftului în punctele de calibrare a) punctul Op; b) punctul A; c) punctul B.

coordonate\puncte	${}^{R}O_p$	${}^{R}A$	${}^{R}B$
x_i	-243.883	-273.710	-397.961
y_i	-348.618	-170.892	-273.635
z_i	-22.440	-21.052	-21.527

Tabelul 6. 2 Coordonatele punctelor necesare pentru calibrarea robotului.

Dat fiind faptul că se cunosc coordonatele celor trei puncte faţă de sistemul ataşat bazei robotului (sistemul „R"), se cunosc şi vectorii de poziţie (${}^{R}\overline{r_{Op}}$, ${}^{R}\overline{r_A}$, ${}^{R}\overline{r_B}$) ale respectivelor puncte faţă de acelaşi sistem.

$$\overline{O_R O_P} = {}^{R}\overline{r_{Op}} = x_1 \cdot \bar{i} + y_1 \cdot \bar{j} + z_1 \cdot \bar{k} \tag{6.14}$$

$$\overline{O_R A} = {}^{R}\overline{r_A} = x_2 \cdot \bar{i} + y_2 \cdot \bar{j} + z_2 \cdot \bar{k} \tag{6.15}$$

$$\overline{O_R B} = {}^{R}\overline{r_{Op}} = x_3 \cdot \bar{i} + y_3 \cdot \bar{j} + z_3 \cdot \bar{k} \tag{6.16}$$

Dacă se cunoaşte vectorul de poziţie al unui punct (M) în raport cu sistemul de referinţă „R", adică ${}^{R}\overline{r_M}$, coordonatele punctului M în raport cu sistemul „P" sunt calculate după formula:

$$\underline{{}^{P}r_M} = \underline{{}^{P}T_R} * \underline{{}^{R}r_M} \tag{6.17}$$

unde

$$\underline{{}^{P}T_R} = (\underline{{}^{R}T_P})^{-1} \tag{6.18}$$

este matricea de trecere de la sistemul de referinţă „R” la sistemul de referinţă „P”, iar ${}^{R}\underline{r_M}$ şi ${}^{P}\underline{r_M}$ sunt matricele transpuse ale vectorului de poziţie al punctului în raport cu sistemul de referinţă „R” şi respectiv „P”. Matricea ${}^{R}\underline{T_P}$ exprimă situarea relativă a sistemului de referinţă „P” faţă de sistemul de referinţă „R”.

Matricea de trecere din sistemul de referinţă „P” în sistemul de referinţă „R” este de forma:

$$ {}^{R}\underline{T_P} = \begin{bmatrix} \cos(x_R, x_P) & \cos(x_R, y_P) & \cos(x_R, z_P) & x_1 \\ \cos(y_R, x_P) & \cos(y_R, y_P) & \cos(y_R, z_P) & y_1 \\ \cos(z_R, x_P) & \cos(z_R, y_P) & \cos(z_R, z_P) & z_1 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{6.19} $$

unde, de exemplu $\cos(y_R, x_P)$ reprezintă cosinusul director al unghiului dintre axa Y a sistemului de referinţă „R” (axa Y_R) şi axa X a sistemului de referinţă „P” (axa X_P) iar x_1, y_1, şi z_1 sunt coordonatele lui Op faţă de sistemul de referinţă „R”.

Pentru determinarea elementelor matricei de trecere ${}^{R}\underline{T_P}$, respectiv cosinuşii directori s-a utilizat un calcul vectorial ce presupune mai multe etape:

1. În primă fază se calculează vectorii $\overline{O_PA}$ şi $\overline{O_PB}$:

$$ \overline{{}^{R}O_PA} = (x_2 - x_1)\cdot\bar{i} + (y_2 - y_1)\cdot\bar{j} + (z_2 - z_1)\cdot\bar{k} = a_1\cdot\bar{i} + b_1\cdot\bar{j} + c_1\cdot\bar{k} \tag{6.20} $$

$$ \overline{{}^{R}O_PB} = (x_3 - x_1)\cdot\bar{i} + (y_3 - y_1)\cdot\bar{j} + (z_3 - z_1)\cdot\bar{k} = a_2\cdot\bar{i} + b_2\cdot\bar{j} + c_2\cdot\bar{k} \tag{6.21} $$

2. Se realizează operaţia de ortogonalizare a vectorilor şi se calculează unghiul α:

$$ \alpha = \frac{-(a_1\cdot a_2 + b_1\cdot b_2 + c_1\cdot c_2)}{a_1^{\,2} + b_1^{\,2} + c_1^{\,2}} \tag{6.22} $$

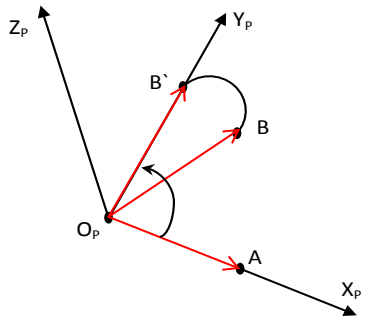


Figura 6. 9: Punctele Op, A şi B în sistemul de referinţă „P”.

1. Se determină vectorul OpB' astfel încât acesta să fie perpendicular pe vectorul OpA:

$$^{R}\overline{O_PB'} = a_3 \cdot \bar{i} + b_3 \cdot \bar{j} + c_3 \cdot \bar{k} = (a_2 + \alpha \cdot a_1) \cdot \bar{i} + (b_2 + \alpha \cdot b_1) \cdot \bar{j} + (c_2 + \alpha \cdot c_1) \cdot \bar{k} \quad (6.23)$$

2. Se verifică perpendicularitatea vectorilor O_PA şi O_PB':

$$a_1 \cdot a_3 + b_1 \cdot b_3 + c_1 \cdot c_3 = 0 \quad (6.24)$$

3. Se determină vectorul O_Pz_P:

$$\overline{^{R}O_Pz_P} = \overline{^{R}O_PA} \times \overline{^{R}O_PB'} = \begin{bmatrix} i & j & k \\ a_1 & b_1 & c_1 \\ a_3 & b_3 & c_3 \end{bmatrix} \quad (6.25)$$

$$^{R}\overline{O_Pz_P} = a_4 \cdot \bar{i} + b_4 \cdot \bar{j} + c_4 \cdot \bar{k}$$

$$^{R}\overline{O_Pz_P} = (b_1 \cdot c_3 - b_3 \cdot c_1) \cdot \bar{i} + (-a_1 \cdot c_3 + a_3 \cdot c_1) \cdot \bar{j} + (a_1 \cdot b_3 - a_3 \cdot b_1) \cdot \bar{k} \quad (6.26)$$

4. Se verifică perpendicularitatea vectorilor O_PA şi O_Pz_P:

$$a_3 \cdot a_4 + b_3 \cdot b_4 + c_3 \cdot c_4 = 0 \quad (6.27)$$

5. Se calculează cosinuşii directori necesari construirii matricei de trecere din sistemul de referinţă „P” în sistemul de referinţă „R”:

$$\cos(x_R, x_P) = \cos(O_R x_R, \overline{O_PA}) = \frac{a_1}{\sqrt{a_1^2 + b_1^2 + c_1^2}} \quad (6.28)$$

$$\cos(y_R, x_P) = \cos(O_R y_R, \overline{O_PA}) = \frac{b_1}{\sqrt{a_1^2 + b_1^2 + c_1^2}} \quad (6.29)$$

$$\cos(z_R, x_P) = \cos(O_R z_R, \overline{O_PA}) = \frac{c_1}{\sqrt{a_1^2 + b_1^2 + c_1^2}} \quad (6.30)$$

$$\cos(x_R, y_P) = \cos(O_R x_R, \overline{O_PB'}) = \frac{a_3}{\sqrt{a_3^2 + b_3^2 + c_3^2}} \quad (6.31)$$

$$\cos(y_R, y_P) = \cos(O_R y_R, \overline{O_PB'}) = \frac{b_3}{\sqrt{a_3^2 + b_3^2 + c_3^2}} \quad (6.32)$$

$$\cos(z_R, y_P) = \cos(O_R z_R, \overline{O_PB'}) = \frac{c_3}{\sqrt{a_3^2 + b_3^2 + c_3^2}} \quad (6.33)$$

$$\cos(x_R, z_P) = \cos(O_R x_R, \overline{O_Pz_P}) = \frac{a_4}{\sqrt{a_4^2 + b_4^2 + c_4^2}} \quad (6.34)$$

$$\cos(y_R, z_P) = \cos(O_R y_R, \overline{O_Pz_P}) = \frac{b_4}{\sqrt{a_4^2 + b_4^2 + c_4^2}} \quad (6.35)$$

$$\cos(z_R, z_P) = \cos(O_R z_R, \overline{O_Pz_P}) = \frac{c_4}{\sqrt{a_4^2 + b_4^2 + c_4^2}} \quad (6.36)$$

6.3. Calibrarea camerei

O altă componentă importantă a standului experimental este camera video LBCKSHL. Această este o cameră analogică, de supraveghere video, color, dotată cu un senzor CCD Sony de tip 1/3, cu o rezoluţie de 540 linii TV.

În această secţiune este prezentată, în detaliu, o descriere a metodei de calibrare utilizată pentru camera video menţionată anterior. Metoda a fost dezvoltată pe baza algoritmului lui Zhengyou Zhang (ZHang, 1999) şi cel a lui Janne Heikkilä (Heikkilä, 1997), fiind utilizată în calibrarea camerei pentru aplicaţia de faţă, deoarece, prezintă un grad de flexibilitate ridicat, faţă de metodele clasice prezentate în capitolul 2. De asemenea a fost aleasă deoarece era deja implementată în Matlab de Jean-Yves Bouguet şi testată cu succes de-a lungul unui număr mare de ani.

Distribuţia gratuită a toolbox-ului de calibrare a camerelor video implementat în Matlab se găsește pe pagina personala a autorului (Bouguet, 2012). Rularea sa, se realizează prin apelarea, în Matlab, a comenzii „calib_gui(0)”, ce permite deschiderea unei interfețe grafice (figura 6.10).

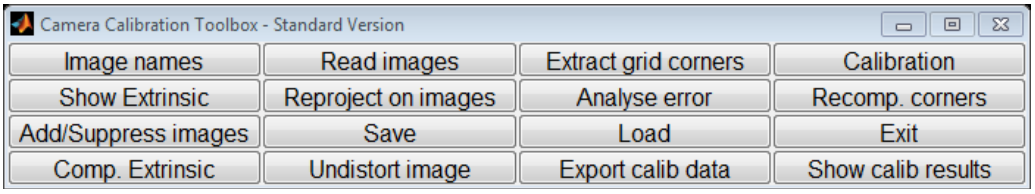


Figura 6. 10: Interfaţa grafică a toolbox-ului de calibrare a camerelor video

Procedura de calibrare începe prin captarea a cel puțin douăzeci şi cinci de imagini (sau mai multe) ale unui şablon de calibrare, de tip tabla de şah (chess-board) care se regăseşte în câmpul vizual al camerei, având situări diferite (poziţii şi orientări diferite). Condiţia de bază impusă la achiziţionarea acestor imagini este aceea ca şablonul menţionat să se regăsească în întregime în imagini.

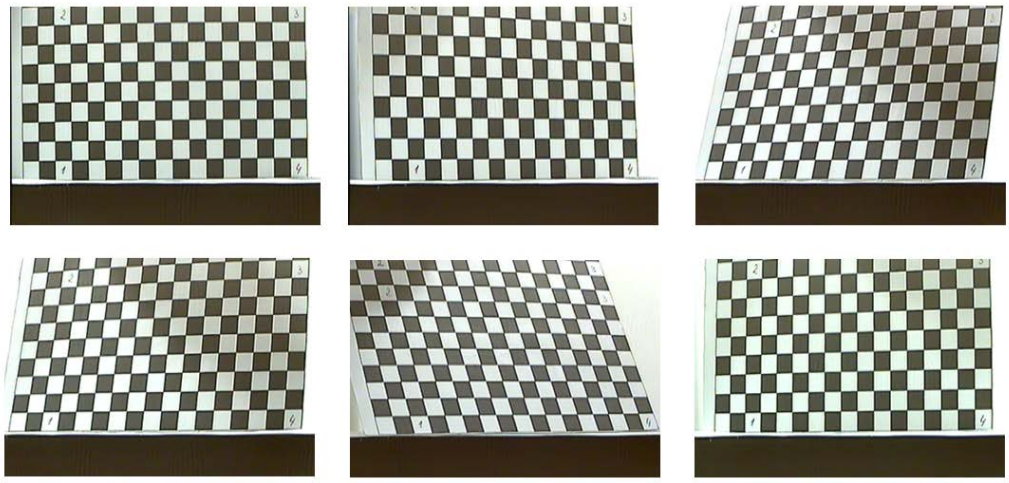


Figura 6. 11: Şase imagini folosite în procesul de calibrare. Numerele indică ordinea de selectare a colţurilor.

Pasul următor constă în extragerea colţurilor şablonului de calibrare prin utilizarea unui detector de trăsături care caută în imagine (în zona de interes) locațiile unde apare o diferenţă mare de contrast. Zona de interes se defineşte prin selectarea manuală în imagine a patru colţuri. Condiţia impusă, este ca primul colţ selectat să fie acelaşi în toate imaginile (deoarece reprezintă originea sistemului de referinţă ataşat imaginilor). Pentru ca operaţia de extragere să se realizeze corect sunt introduse ca date iniţiale dimensiunea zonei de căutare (în pixeli) şi dimensiunile pătratelor şablonului (în milimetri). Toolbox-ul este prevăzut cu o funcţie de numărare automată a pătratelor şablonului. Rezultatele obţinute în urma acestui pas sunt ilustrate în figura 6.12, pe una din imaginile folosite.

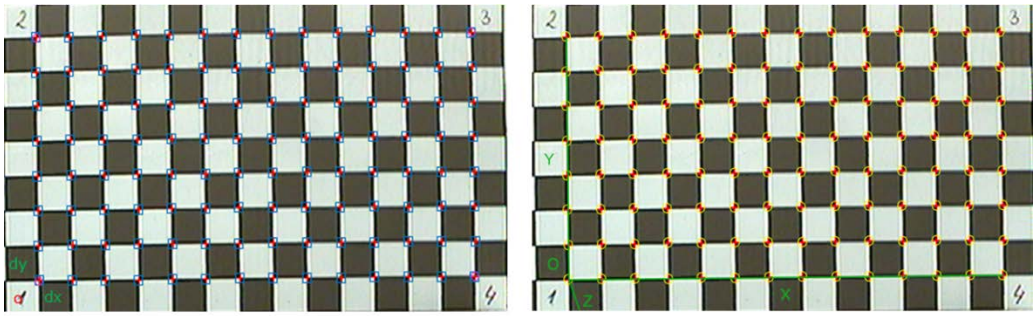


Figura 6. 12: Zona de interes a unei imagini utilizate în calibrare: a) puncte (colţuri) extrase; b) sistem de coordonate ataşat imaginii.

În urma acestei etape, sunt stocate coordonatele fiecărei trăsături, faţă de sistemul de referinţă al imaginii şi cel al obiectului (al şablonului). Pe baza coordonatelor fiecărui punct cunoscut atât în imagine, cât şi în realitate, se rezolvă un sistem liniar de ecuaţii cu zece necunoscute. Aceste necunoscute sunt (Bouguet, 2012):

- Distanţa focală (în pixeli): vectorul 2x1 *fc(*fc_x*,* fc_y*)*;
- Coordonatele centrului imagini sau punctului principal: vectorul 2x1 *cc(*cc_x*,* cc_y*)*;
- Unghiul de înclinare (skew angle) ce reprezintă unghiul dintre axa x şi y a elementelor de imagine a senzorului vizual (în pixeli): scalarul *alpha_c*.
- Coeficienţii de distorsiune (distorsiunea radială şi cea tangenţială): vectorul 5x1 *kc*.

Modelul matematic poate fi sintetizat după cum urmează:

1. Se consideră un punct „P" din spaţiu, al cărui vector de poziţie faţă de sistemul de referinţă ataşat camerei video este:

$$XX_C = [X_C; Y_C; Z_C] \tag{6.37}$$

2. Se proiectează punctul în planul imagine utilizând parametrii intrinseci ai camerei (fc, cc, alpha_c şi kc).
3. Se consideră x_n ca fiind proiecția normalizată în cazul unei camere ideale (pinhole camera);

$$x_n = \begin{bmatrix} \frac{X_C}{Z_C} \\ \frac{Y_C}{Z_C} \end{bmatrix} = \begin{bmatrix} x \\ y \end{bmatrix} \tag{6.38}$$

4. Se consideră $r^2 = x^2 + y^2$;
5. Se includ distorsiunile de imagine astfel încât noul punct normalizat se defineşte ca:

$$x_d = \begin{bmatrix} x_d(1) \\ x_d(2) \end{bmatrix} = \left(1 + kc(1)r^2 + kc(2)r^4 + kc(5)r^6\right)x_n + dx \tag{6.39}$$

unde *dx* este vectorul distorsiunii tangenţiale.

6. Se determină coordonatele în pixeli $x\_pixel = [x_p; y_p]$ a proiecţiei punctului „P” în planul imagine;

$$\begin{cases} x_p = fc(1)\left(x_d(1) + alpha\_c * x_d(2)\right) + cc(1) \\ y_p = fc(2)x_d(2) + cc(2) \end{cases} \tag{6.40}$$

Conform ecuaţiei 6.40 se observă, că între coordonatele în pixeli şi coordonatele normalizate ale punctului „P”, există o relaţie de legătură pe baza unei ecuaţii liniare:

$$\begin{bmatrix} x_p \\ y_p \\ 1 \end{bmatrix} = KK \begin{bmatrix} x_d(1) \\ x_d(2) \\ 1 \end{bmatrix} \tag{6.41}$$

unde KK este matricea camerei ce se defineşte astfel:

$$KK = \begin{bmatrix} fc(1) & alpha\_c * fc(1) & cc(1) \\ 0 & fc(2) & cc(2) \\ 0 & 0 & 1 \end{bmatrix} \tag{6.42}$$

În urma primei rulării a funcţiei de calibrare, se obţin, ca rezultate, un set de valori estimative pentru parametri intrinseci. Pentru a obține o determinare mai exactă, respectiv pentru reducerea erorilor de pixeli, matricea coordonatelor fiecărei trăsături din imagine se poate reproiecta. Toolbox-ul permite, de asemenea, o analiză a erorilor de pixeli, ce sunt introduse de fiecare trăsătură extrasă din setul de imagini analizat şi reproiectată (vezi figura 6.13). Pe baza acesteia, se poate identifica trăsătura sau trăsăturile (împreună cu imaginea corespunzătoare) care

introduc cele mai multe erori, astfel încât ele pot fi excluse sau din nou extrase, iar procedura de calibrare poate fi încă o dată executată.

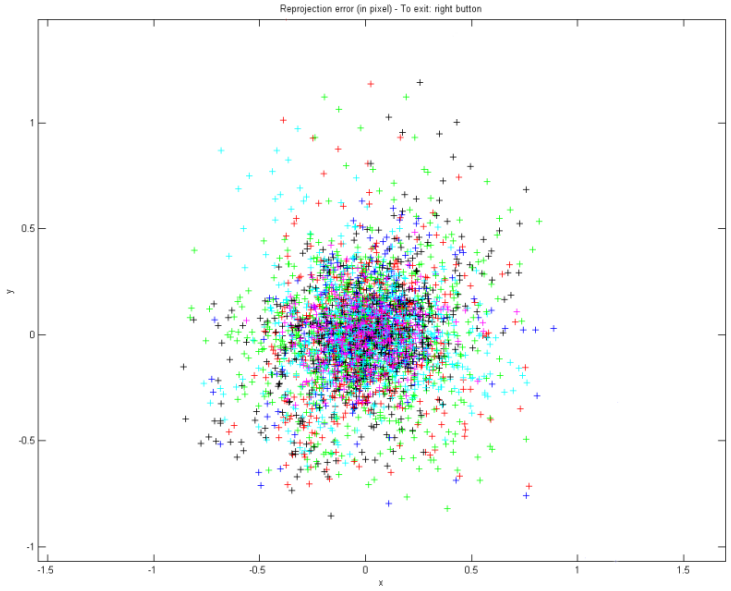


Figura 6. 13: Reprezentare grafică a erorilor de re-proiecţie de pixeli

Se observă, că erorile de reproiectare a trăsăturilor, pentru camera video LBCKSHL, au valoare maximă de aproximativ 1,5 pixeli.

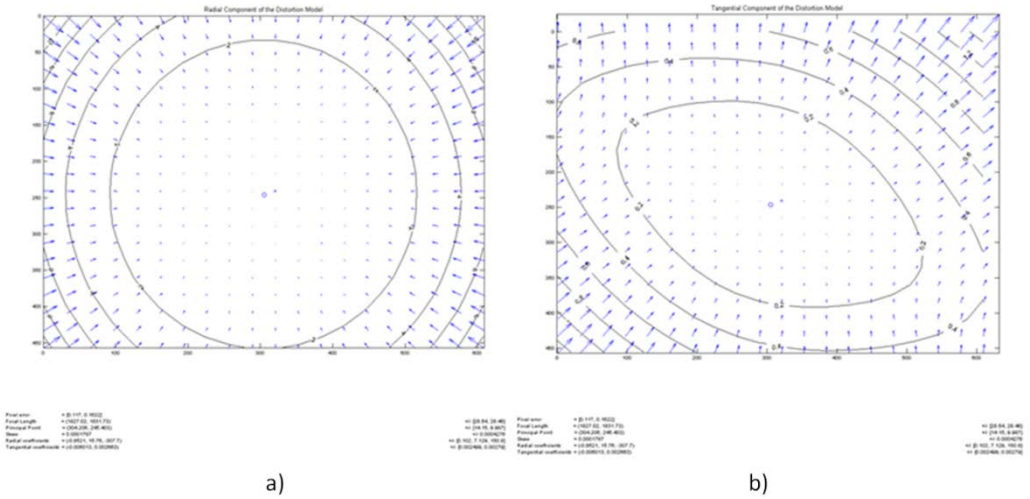
Etapa de reproiectare şi reextragere a trăsăturilor din imagine permite reducerea erorilor de calibrare per ansamblu, respectiv sunt reduse de asemenea şi erorile de estimare a coeficienţilor de distorsiune şi a unghiului de înclinare.

Valorile asociate parametrilor intrinseci, obţinute în urma operaţiei de calibrare a camerei video utilizată în această aplicaţie sunt prezentate în tabelul 6.3.

Parametri intrinseci	Valori
Distanţa focală	fc = [1827.0245 1831.7264] ± [6.50753 6.22780]
Punctul principal	cc = [304.2061 245.4029] ± [1.89402 1.44237]
Unghiul de înclinare	alpha_c = [0.00018] ± [0.00009]
Coeficienţii de distorsiune	kc = [-0.8521 15.7573 -0.0060 0.0026 -307.6518] ± [0.10683 1.27153 0.00391 0.00527 1.69717]
Erori de pixeli	err = [0.1170 0.1622]

Tabelul 6. 3 Parametri intrinseci ai camerei video LBCKSHL.

Pe baza parametrilor obţinuţi, se poate realiza şi o reprezentare grafică a efectului distorsiunilor introduse de obiectivul camerei video asupra pixelilor din imagini (figura 6.14). În prima figură, se prezintă impactul pe care îl au distorsiunile radiale asupra fiecărui pixel al imaginii. Fiecare săgeată indică deplasarea efectivă a unui pixel, indusă de componenta distorsiunilor radiale ale lentilei. Se poate observa, că punctele de la colţurile imaginii sunt deplasate cu aproximativ 10 pixeli. Cea de-a doua figură ilustrează efectul pe care îl produce componenta tangenţială a distorsiunilor asupra imagini. Influenţa acesteia este redusă, putând fi chiar ignorată deoarece deplasarea maximă este doar de 1.2 pixeli (colţul dreapta sus).



a) b)

Figura 6. 14: Reprezentarea grafică a distorsiunilor asupra imagini: a) componenta radială; b) componenta tangenţială.

Se observă, de asemenea, că algoritmul de calcul implementat în toolbox permite şi o estimare a valorilor numerice, ce corespund erorilor de calibrare. Acestea sunt aproximativ de trei ori deviaţia standard a estimării.

În vederea obţinerii unor erori reduse de calibrare, se aplică o regulă, acceptată în general, care subliniază faptul că pentru o anumită precizie de calibrare este necesar ca şablonul de calibrare să fie proiectat având o toleranţă cu una sau două ordine de mărime mai mică.

Algoritmul iterativ, implementat în toolbox, permite utilizarea întregului număr de imagini, ce conţin şablonul de tip „tabla de şah", folosit la calibrarea camerei, şi coordonatele colţurilor acestuia, pentru a determina situarea sistemului de referinţă ataşat şablonului faţă de sistemul de referinţă al camerei video.

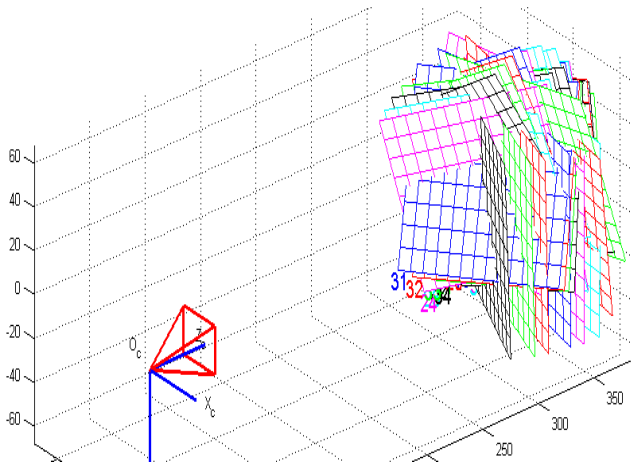


Figura 6. 15: Situările şablonului de calibrare faţă de sistemul de referinţă al camerei video.

Pe baza parametrilor intrinseci ai camerei video LBCKSHL, şi a unei imagini cu şablonul de calibrare situat în planul obiect, corespunzător planului în care s-a

realizat calibrarea robotului SCORA ER 14 (a se vedea subcapitolul 6.1.1) se pot determina parametri extrinseci ai camerei. Aceştia sunt necesari relaţionării sistemului de referinţă al camerei video, sistemul $O_cX_cY_cZ_c$, cu sistemul de referinţă ataşat şablonului de calibrare $O_{P'}X_{P'}Y_{P'}Z_{P'}$.

Parametri extrinseci	Valori
Vectorul de translaţie [mm]	Tc_ext = [-52.23602 11.61013 419.13637]
Vectorul de rotaţie [rad]	omc_ext = [3.13531 -0.01384 -0.046763]
Matricea de rotaţie [rad]	Rc_ext = [0.99951 -0.00874 -0.02984 -0.00891 -0.99994 -0.00576 -0.02979 0.00602 -0.99953]
Erori de pixeli	err = [0.17368 0.17228]

Tabelul 6. 4 Parametri extrinseci ai camerei video LBCKSHL.

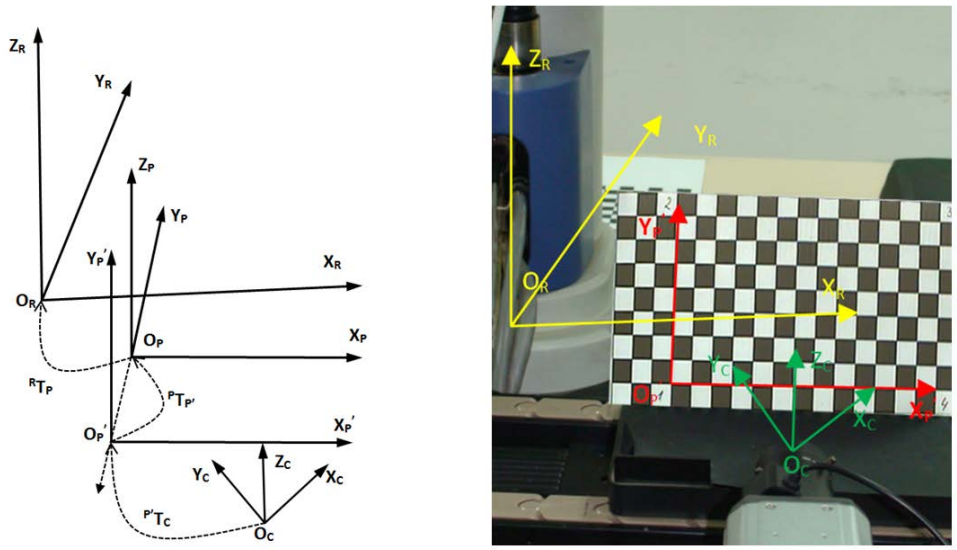


Figura 6. 16: Trecerea de la sistemul de referinţă al camerei la sistemul de referinţă ataşat robotului.

Trecerea de la sistemul de referinţă al camerei la sistemul de referinţă ataşat şablonului de calibrare se face, pentru un punct M, astfel:

Se consideră punctul M având vectorul de poziţie $XX_{P'} = \left[X_{P'}, Y_{P'}, Z_{P'}\right]$ faţă de sistemul $O_{P'}X_{P'}Y_{P'}Z_{P'}$ şi vectorul de poziţie $XX_C = \left[X_C, Y_C, Z_C\right]$ faţă de sistemul $O_CX_CY_CZ_C$.

Relaţia de legătură între cei doi vectori este:

$$XX_C = Rc\_ext * XX_{P'} + Tc\_ext \tag{6.43}$$

Această metodă de calibrare a camerei video, respectiv de determinare a parametrilor intrinseci şi extrinseci, a fost utilizată pentru toate aplicaţiile de vedere robotizată, realizate în Matlab, ce sunt prezentate în teza de faţă.

6.4. Aplicaţia propriu-zisă

În acest paragraf, se prezintă aplicaţia dezvoltată în Matlab, ce presupune manipularea controlată a unor obiecte colorate. Aplicaţia este de tip servoing vizual, ce se bazează pe imagini, denumită şi aplicaţie de tip 2D (Image – based Visual Servoing sau IBVS). Acest lucru, se datorează faptului că, controlul mişcării robotului se realizează utilizând informaţiile de natură 2D extrase din imagini ce sunt prelevate de o singura cameră a cărei situare este cunoscută. Poziţia şi orientarea camerei faţă de un sistem de coordonate cunoscut (în cazul de faţă, sistemul de coordonate ataşat spaţiului de lucru sau spaţiul piesei), s-a determinat în urma operaţiei de calibrare a camerei video conform modelului prezentat în secţiunea 6.1.2. De asemenea, situarea sistemului de referinţă ataşat spaţiului piesei, faţă de sistemul de referinţă ataşat bazei robotului a fost determinată (vezi secţiunea 6.1.1). Standul experimental creat, pe care s-a implementat aplicaţia, este prezentat schematic în figura 6.17 (Pop, 2012a).

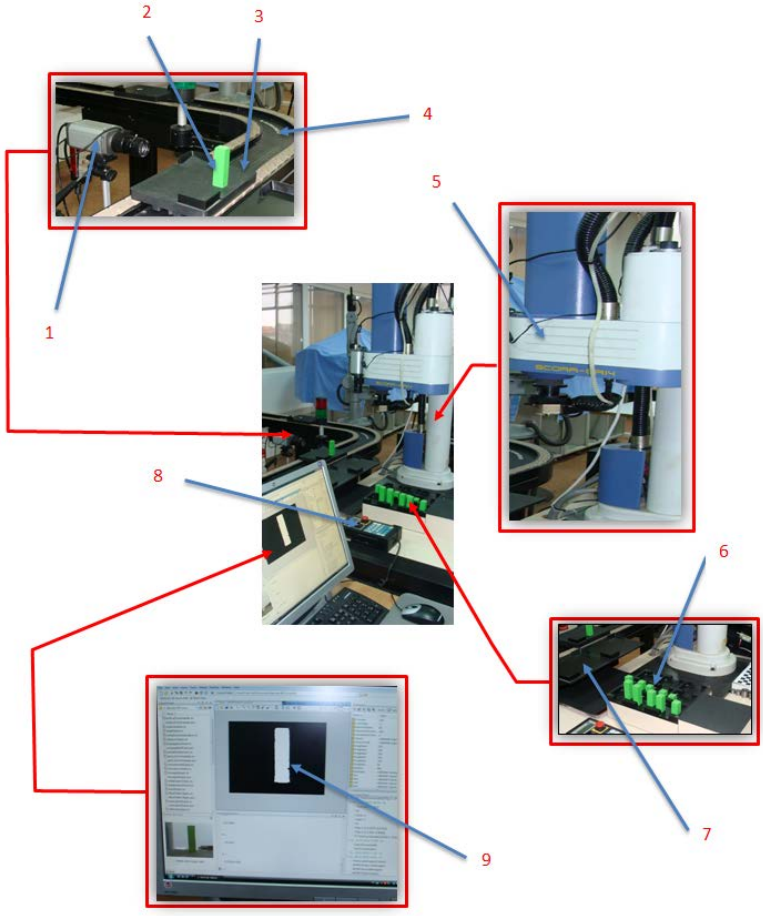
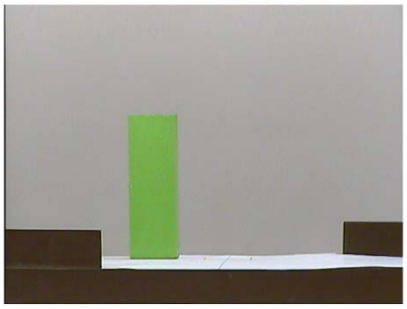
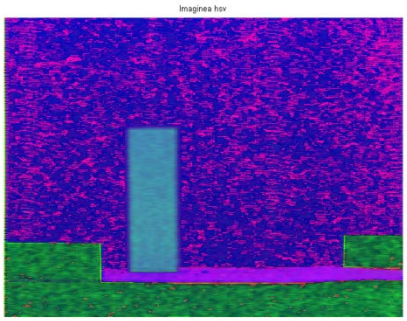


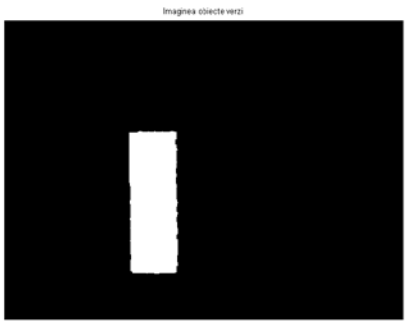
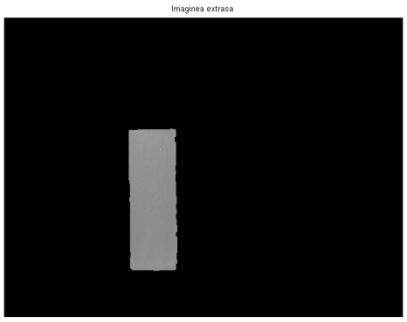
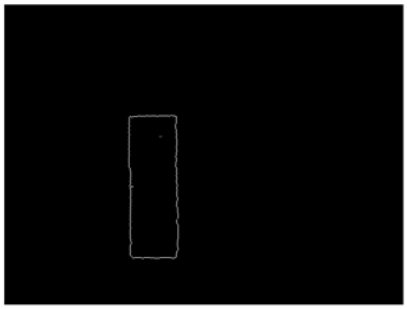
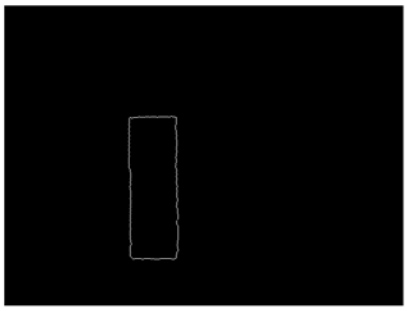
Figura 6. 17: Standul experimental : 1) camera video CCD; 2) obiectul vizat; 3) cărucior; 4) conveior de transfer; 5) robotul SCORA ER 14; 6) depozit de piese; 7) post de aşteptare; 8) teach pendant; 9) calculatorul pe care rulează aplicaţia

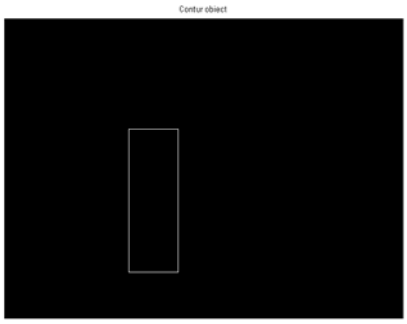
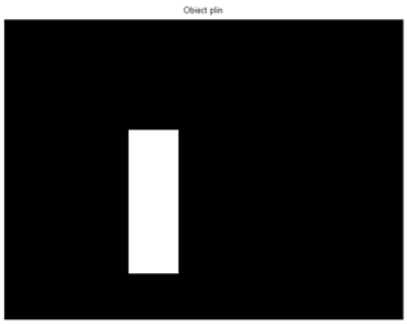
Aplicaţia constă practic, în determinarea poziţiei de prindere a unui obiect colorat (un paralelipiped dreptunghic verde), ce sosește pe căruciorul unui conveior de transfer la staţia de lucru a robotului SCORA ER 14. Poziţia piesei pe cărucior este cunoscută (X_P şi Y_P cunoscut), fiind constantă, astfel încât pe baza algoritmului de vedere computerizată dezvoltat, se determină din imagini doar poziția de prindere a obiectului (poziţia de închidere a bacurilor dispozitivului de prehensare). Piesele în cauză, variază în funcţie de înălţime, iar poziţia de prindere se găsește la jumătatea înălțimii obiectului. Prinderea se realizează în acest mod, pentru a avea o manipulare controlată a piesei, din momentul în care se prelevează de pe cărucior, până în momentul în care se depune în locația dorită.

Algoritmul de vedere computerizată, care face posibilă realizarea acestei aplicații, presupune o segmentare a imaginii pe bază de culoare. Această operație de prelucrare a imaginii se realizează în spațiul de culoare HSV (Hue, Saturation, Value sau Nuanţă, Saturaţie, Valoare) deoarece este un sistem monocromatic, fapt ce face ca o singură componentă să influenţeze culoarea dominantă. Celelalte componente se referă la intensitatea culorii şi la strălucirea relativă.

Etapele principale pe care le presupune algoritmul de prelucrare a imaginii (anexa 7) sunt prezentate în tabelul 6.5.

Etapa		Rezultat
1	Achiziţia imaginii în formatul .png (Portable Network Graphics) la o rezoluţie de 640x480 pixeli, în spaţiul de culoare RGB (Red, Green, Blue - sistem tricromatic aditiv - Roşu, Verde, Albastru).	
2	Conversia imaginii în spaţiul de culoare HSV cu ajutorul funcției „rgb2hsv". Această etapă se execută pentru a se detecta mai ușor şi mai precis (cu erori de pixeli mai mici) obiectul vizat în imagine. Detectarea se realizează pe baza unor valori de prag (limita minimă şi limita maximă) ale celor trei componente. Valorile de prag se obţin, într-o etapă premergătoare rulării aplicaţiei. Această etapă constă în selectarea manuală, în mai multe imagini, a unui număr de pixeli ce aparţin obiectului căutat. Pe baza valorilor acestor pixeli sunt determinate limitele de prag utilizate de algoritm.	

3	Detectarea pe baza limitelor de prag a obiectului vizat.	
4	Convertirea imaginii originale în nuanţe de gri şi extragerea din imaginea rezultantă a pixelilor ce aparţin obiectului detectat în etapă anterioară.	
5	Detectarea muchiilor pe baza operatorului Sobel. În fiecare punct al imaginii, operatorul Sobel calculează opusul gradientului intensităţii imaginii.	
6	Eliminarea obiectelor mici. Sunt eliminate acele obiecte al căror număr de pixeli este mai mic decât pragul stabilit. Pragul se stabilește pe baza numărului de pixeli corespunzători celui mai mic obiect vizat.	

7	Rectificarea conturului. Acest lucru se realizează prin determinarea coordonatelor celor patru colţuri ale conturului: Colţul stânga sus (x_{minim}, y_{minim}); Colţul stânga jos (x_{minim}, y_{maxim}); Colţul dreapta jos (x_{maxim}, y_{maxim}); Colţul dreapta sus (x_{maxim}, y_{minim}); Pe baza acestor coordonate se generează grafic liniile noului contur utilizând funcţia special dezvoltată „linie”.	
8	Obţinerea obiectului final. Acesta reprezintă corespondentul cel mai fidel al obiectului real în pixeli. Pe baza lui se pot determina caracteristicile geometrice ale obiectului real.	

Tabelul 6. 5: Etapele algoritmului de prelucrare a imagini cu obiectul vizat.

Algoritmul de prelucrare de imagini prezentat anterior este integrat împreună cu informaţiile obţinute în urma proceselor de calibrare a camerei video şi a robotului în algoritmul de vedere computerizată. Implementarea acestui pe standul experimental existent, conduce la realizarea aplicaţiei de servoing vizual.

Ordinograma (diagrama de flux) aplicației este ilustrată în figura 6.18. Paşii ce trebuie parcurşi pentru realizarea aplicaţiei sunt următorii:

Pasul 1. Obiectul (unul din paralelipipedele dreptunghice de culoare verde) este plasat pe căruciorul conveior-ului de transport într-o poziţie fixă (cunoscută);

Pasul 2. Căruciorul se deplasează până la staţia de lucru (staţia robotului SCORA ER 14) şi se opreşte în faţa camerei video;

Pasul 3. Camera CCD achiziţionează o imagine cu obiectul aflat pe cărucior la o distanţă cunoscută faţă de cameră;

Pasul 4. Algoritmul de procesare de imagini (prezentat anterior) prelucrează imaginea achiziţionată. Rezultatul acestuia este o imagine binară cu obiectul detectat.

Pasul 5. Se determină aria obiectului în pixeli (A_{pix}) pe baza imaginii obţinută în pasul anterior.

Pasul 6. Se determina lăţimea pieselor în pixeli (l_{pix}) şi ulterior factorul de scalare S (raportul pixel / milimetru). Acesta se poate determina avându-se în vedere faptul că singura dimensiune de gabarit care variază la toate piesele este înălţimea (lungimea dreptunghiului din imagini). Lăţimea pieselor este constantă şi se cunoaşte în milimetri (lmm) (grosimea nu

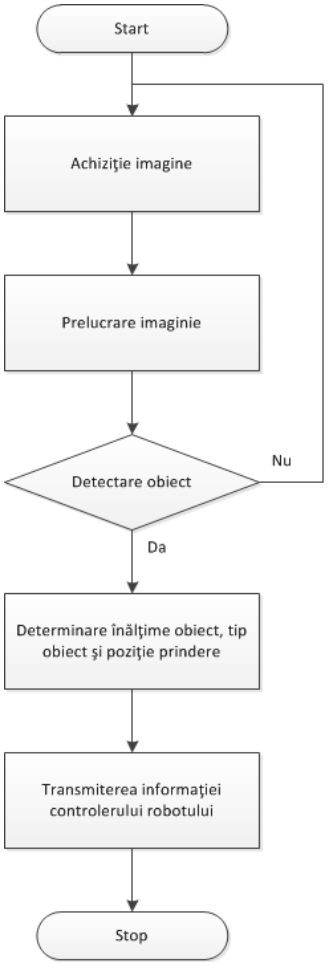


Figura 6. 18: Ordinograma aplicaţiei de determinare a înălţimii, tipului şi coordonatelor de prindere a obiectelor colorate.

prezintă interes, se neglijează pentru aplicaţia de faţă).

$$S = \frac{l_{pix}}{l_{mm}} \tag{6.44}$$

Pasul 7. Se determină înălţimea în pixeli (H_{pix}):

$$H_{pix} = \frac{A_{pix}}{l_{pix}} \tag{6.45}$$

Pasul 8. Se calculează înălţimea obiectului detectat în milimetri pe baza factorului de scalare:

$$H_{mm} = H_{pix} * \frac{1}{S} \tag{6.46}$$

Pasul 9. Se determină tipul de obiect pe baza înălţimii obţinute în pasul anterior.

Pasul 10. Se calculează coordonata pe axa OZ faţă de planul piesei, pentru a se putea realiza prinderea obiectului. Reprezintă jumătate din înălțimea calculată.

Pasul 11. Se transformă coordonatele de prindere ale obiectului, ce sunt cunoscute faţă de sistemul ataşat planul piesei (x, y constante şi z determinat) în coordonatele faţă de sistemul ataşat robotului utilizând matricea de trecere ${}^{R}T_{P}$.

Pasul 12. Se comandă robotul pe baza rezultatele de la pasul 9 şi 11.

Pasul 13. Robotul pe baza informaţiilor primite, prelevează piesa de pe cărucior şi o depozitează în postul de aşteptare sau în locaşul corespunzător din depozitul de piese (anexa 8).

6.5. Concluzii

După efectuarea unui număr mare de teste experimentale pe obiecte de diferite dimensiuni (diferența de înălțime este de cel puțin 5 mm), s-a observat că sistemul, în configurația prezentată, este capabil să detecteze, localizeze și identifice obiectele într-un mod corespunzător.

Numărul obiectului	Tipul obiectului	Valoarea numerică a înălţimii măsurată [mm]	Tipul obiectului determinat	Valoarea numerică a înălţimii calculată [mm]
1	A	18.37	A	18.66
2	B	23.40	B	23.79
3	C	28.48	C	28.90
4	D	33.50	D	33.94
5	E	38.56	E	38.73
6	F	43.49	F	43.88
7	G	48.52	G	48.78
8	H	53.55	H	54.04
9	I	58.60	I	58.87
10	J	63.47	J	63.77

Tabelul 6. 6: Rezultate obţinute în urma rulării aplicaţiei pe toate obiectele

Valorile măsurate din tabel, au fost obținute cu un șubler digital, și valorile calculate cu ajutorul relaţiei 6.46. În ciuda rezultatelor satisfăcătoare ce au fost obţinute, sistemul poate fi influențat de diverși factori externi și, prin urmare, ocazional pot să apară erori sistematice care afectează performanța acestuia.
Unele dintre aceste surse de eroare sunt lumină ambientală, coeficientul de reflexie a suprafeţei obiectului, umbre şi balansul de alb.

Sursa de eroare cea mai importantă este lumina ambientală care poate influența semnificativ camera CCD și, prin urmare, imaginile obținute, făcând rezultatele obținute în urma prelucrării lor să dețină erori nedorite. Din acest motiv, se recomandă ca algoritmul prezentat să fie utilizat într-un mediu cu iluminare controlată.

7. CONCEPEREA ŞI DEZVOLTAREA UNOR APLICAŢII INDUSTRIALE DE TIP „PICK AND PLACE" ÎN MATLAB

În acest capitol, sunt prezentate trei aplicații de vedere robotizată, al căror obiectiv principal, este cel de prelevare, manipulare şi depozitare, cu ajutorul unui robot, a unor obiecte de tip industrial. Gradul de complexitate al acestora variază, datorită modificării condițiilor iniţiale impuse fiecărei aplicații în parte, respectiv datorita cantităţii de informație cunoscute.

7.1. Aplicaţie de detectare şi recunoaştere a tipului de obiect aflat într-o poziţie fixă, cunoscută

Scopul aplicaţiei: Detectarea şi recunoaşterea automată a obiectelor ce sosesc pe căruciorul unui conveior de transfer într-o poziţie fixă, cunoscută, pe bază unui algoritm de prelucrări de imagini, pentru sortarea şi stivuirea acestora, în locaşurile corespunzătoare, dintr-un depozit de piese (Pop, 2012b).

Echipamentele utilizate pentru realizarea acestei aplicații sunt:

- Sistem robotizat de manipulare, ce are în componenţă următoarele:
  - Robot de tip scara (SCORA ER 14);
  - Controler central şi controler robot (tip B);
  - Calculator central şi calculator de staţie;
  - Depozit de piese;
  - Cutie de conexiuni;
  - Teach pendant.
- Conveior de transfer ce presupune:
  - Patru cărucioare;
  - Trei posturi de aşteptare;
  - Un PLC.
- Sistem de vedere artificial compus din:
  - Două camere video CCD de supraveghere (de tip LBCKSHL). Una montata pe un trepied într-o poziţie fixă, cunoscută şi ce-a de-a doua montată pe axa 3 a robotului SCORA;
  - Doua plăci de captură semnal video;
  - Un calculator (pentru dezvoltarea şi rularea algoritmului de vedere computerizata);
  - Programul de bază Matlab (mediul de lucru în care s-a dezvoltat aplicaţia).
- Un şablon de calibrare 2D de tip „tablă de şah";
- Piese industriale (rulmenți radiali cu bile).

Aplicaţia realizează identificarea, prelevarea, manipularea, sortarea şi depozitarea unor obiecte de tip rulment radial cu bile, de dimensiuni diferite (vezi tabelul 7.1), care sosesc la staţia de lucru pe căruciorul unui conveior de transfer. Configuraţia standul experimental utilizat în acest scop este prezentat în figura 7.1.

Nr. Crt.	Serie rulment	Diametru interior d[mm]	Diametru exterior D[mm]	Laţime B[mm]
1	607	5.9	18.96	5.94
2	608	7.9	21.96	6.94
3	6000	9.9	25.96	7.94
4	6001	11.9	27.96	7.94
5	1602	14.9	31.96	7.9
6	7201	11.9	31.96	9.9
7	6003	16.9	34.96	9.9
8	6204	19.95	46.96	13.95
9	3056204	19.95	46.96	20.55

Tabelul 7. 1: Rulmenţii utilizaţi în cadrul aplicaţiei.

Figura 7. 1: Standul experimental 1) calculatorul pe care rulează aplicaţia; 2) camera CCD 2; 3) căruciorul conveierului; 4) piesa - rulmentul; 5) buffer; 6) camera CCD 1; 7) robotul SCORA-ER14; 8) depozitul de piese; 9) teach pendant.

Etapele premergătoare realizării acestei aplicaţii sunt calibrarea robotului SCORA (vezi subcapitolul 6.1.1) şi calibrarea camerelor video (vezi subcapitolul 6.1.2). Pentru realizarea operaţiei de calibrare a camerelor video, în cadrul acestei aplicaţii s-a utilizat un şablon de calibrare 2D. Metoda de calibrare prezentată în capitolul 6, a fost utilizată şi în acest caz, deoarece fiecare cameră vizualiza individual un plan al şablonului de calibrare. Parametri camerelor (intrinseci si extrinseci) obţinuţi în urma operaţiei de calibrare a camerelor sunt prezentaţi în tabelul 7.2 şi coeficienţii de distorsiune totală (radială şi tangenţială) a lentilelor în figura 7.2.

	Parametri intrinseci	**Valori**
Camera 1	Distanţa focală [pixel]:	fc = [908.03825 992.41205]
	Punctul principal [pixel]:	cc = [363.50768 221.79380]
	Unghiul oblic [grade]:	alpha_c = [-0.00098]
	Distorsiunile:	kc = [-0.30845 0.49046 0.00234 -0.00446 -1.09196]
	Erori de pixeli[pixel]:	err = [0.09427 0.09722]
	Parametri extrinseci	**Valori**
	Vectorul de translaţie [mm]:	Tc_ext = [74.07689 35.99206 340.89596]
	Vectorul de rotaţie [rad]:	omc_ext = [-1.965117 2.447803 0.031526]
	Matricea de rotaţie [rad]:	Rc_ext = [-0.216251 -0.976279 0.010688 -0.976231 0.216052 0.017175 -0.014459 0.014148 -0.999795]
	Erori de pixeli [pixel]:	err = [0.11184 0.09633]
Camera 2	**Parametri intrinseci**	**Valori**
	Distanţa focală [pixel]:	fc = [1793.79262 1797.81346]
	Punctul principal [pixel]:	cc = [281.27936 232.61122]
	Unghiul oblic [grade]:	alpha_c = [0.00044]
	Distorsiunile:	kc = [-0.63457 -0.24512 -0.00607 0.00541 -8.87892]
	Erori de pixeli[pixel]:	err = [0.12040 0.16944]
	Parametri extrinseci	**Valori**
	Vectorul de translaţie [mm]:	Tc_ext = [-49.67934 14.00653 410.43727]
	Vectorul de rotaţie [rad]:	omc_ext = [3.135123 -0.014208 -0.055588]
	Matricea de rotaţie [rad]:	Rc_ext = [0.999330 -0.008955 -0.035476 -0.009166 -0.999941 -0.005783 -0.035422 0.006104 -0.999354]
	Erori de pixeli [pixel]:	err = [0.21634 0.19381]

Tabelul 7. 2: Parametri intrinseci şi extrinseci ai celor două camere CCD.

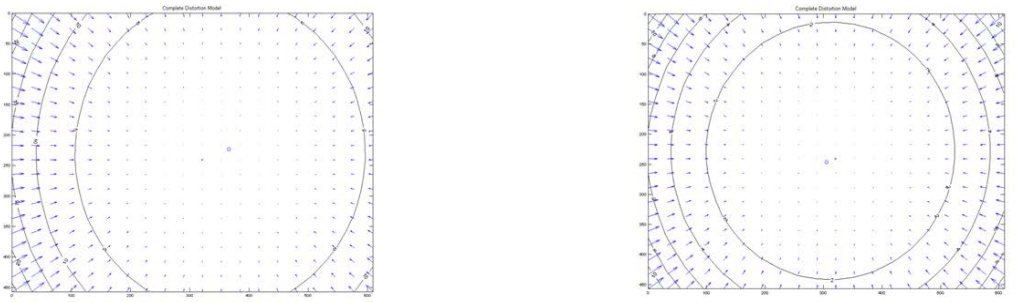


Figura 7. 2: Distorsiunile lentilelor celor două camere CCD.

Această aplicaţie are la bază trei algoritmi de procesare de imagini. Doi dintre aceştia sunt de recunoaştere de obiecte iar al treilea este de identificare.

Modalitatea de abordare a acestei aplicaţii constă în:

- detectarea şi identificarea tipului de rulment pe baza imaginilor achiziţionate de camera CCD 2 (camera de pe trepied fixă);

- detectarea centrul de greutate şi determinarea poziţiei sale pe baza imaginilor prelevate de camera CCD 1 (camera de pe robot).

Pentru realizarea acestei aplicaţii de vedere robotizată s-a ţinut cont de următoarele premise:

- Datele de intrare:
  - Pe cărucior soseşte la staţia de lucru un singur rulment;
  - Situarea rulmenţilor pe cărucior este cunoscută (distanţa de la camera video fixă până la obiect este constantă indiferent de diametrul exterior al rulmentului, respectiv de tipul rulmentului);
  - Senzorii camerelor video sunt paraleli cu planul $O_pX_pY_p$ şi $O_pY_pZ_p$ (proiecţie normală);
  - Pragurile necesare algoritmului de detectare a obiectelor, în spațiul de culoare HSV (Hue, Saturation, Value sau Nuanţă, Saturaţie, Valoare) a imagini prelevată de camera 2 (fixă) sunt cunoscute (se obţin într-o operație anterioare de prelucrare);
  - Şabloanele necesare algoritmului de identificare a tipului de rulment sunt obţinute tot într-o etapă premergătoare aplicaţiei. Acestor şabloane le sunt asociate caracteristicile geometrice ale rulmenţilor (vezi tabelul 7.2).
- Datele de ieşire:
  - Tipul de rulment aflat pe cărucior în momentul rulării aplicaţiei;
  - Poziţia rulmentului depozitat în postul de aşteptare (buffer).

Ordinograma aplicației este ilustrată în figura 7.3. Paşii ce sunt necesari a fi parcurşi pentru realizarea aplicaţiei sunt următorii:

Pasul 1. Un rulment ajunge în faţa camerei CCD 2 pe un cărucior al conveiorului de transfer, într-o poziţie cunoscută;

Pasul 2. Camera CCD 2 achiziţionează o imagine cu rulmentul situat pe cărucior;

Pasul 3. Pe baza imaginii prelevate de camera CCD 2, primul algoritm de procesare a imaginii, detectează prezenţa rulmentului pe cărucior şi conturul acestuia;

Pasul 4. Cel d-al doilea algoritm de procesare a imaginii compară conturul detectat cu un număr de şabloane învăţate în etapa off-line, premergătoare acestei aplicații, determină tipul de rulment şi prin urmare, caracteristicile sale geometrice (inclusiv lăţimea „B” a rulmentului);

Pasul 5. Calculatorul pe care rulează aplicaţia, comandă robotul, pentru ca acesta să se deplaseze dintr-o poziţie de aşteptare într-o poziţie pentru apucarea rulmentului de pe paletă;

Pasul 6. Robotul realizeaza operaţia de prelevare a rulmentului de pe cărucior, respectiv prinde rulmentul între degetele dispozitivului de prehensiune. Se deplasează, după aceea, deasupra postului de aşteptare (buffer-ului staţiei de lucru), deschide dispozitivul de prehensiune şi dă drumul rulmentului să cadă. Ulterior se retrage într-o poziția stabilită pentru a putea preleva o imagine clară a întregului buffer cu ajutorul camerei CCD 1 (fixată pe robot);

Pasul 7. Camera video 1 achiziţionează o imagine cu vederea de sus a rulmentului;

Pasul 8. Cel de-al treilea algoritm de prelucrare de imaginii, detectează, pe baza imagini de la camera 1, centrul de greutate al rulmentului (coordonatele x şi y);

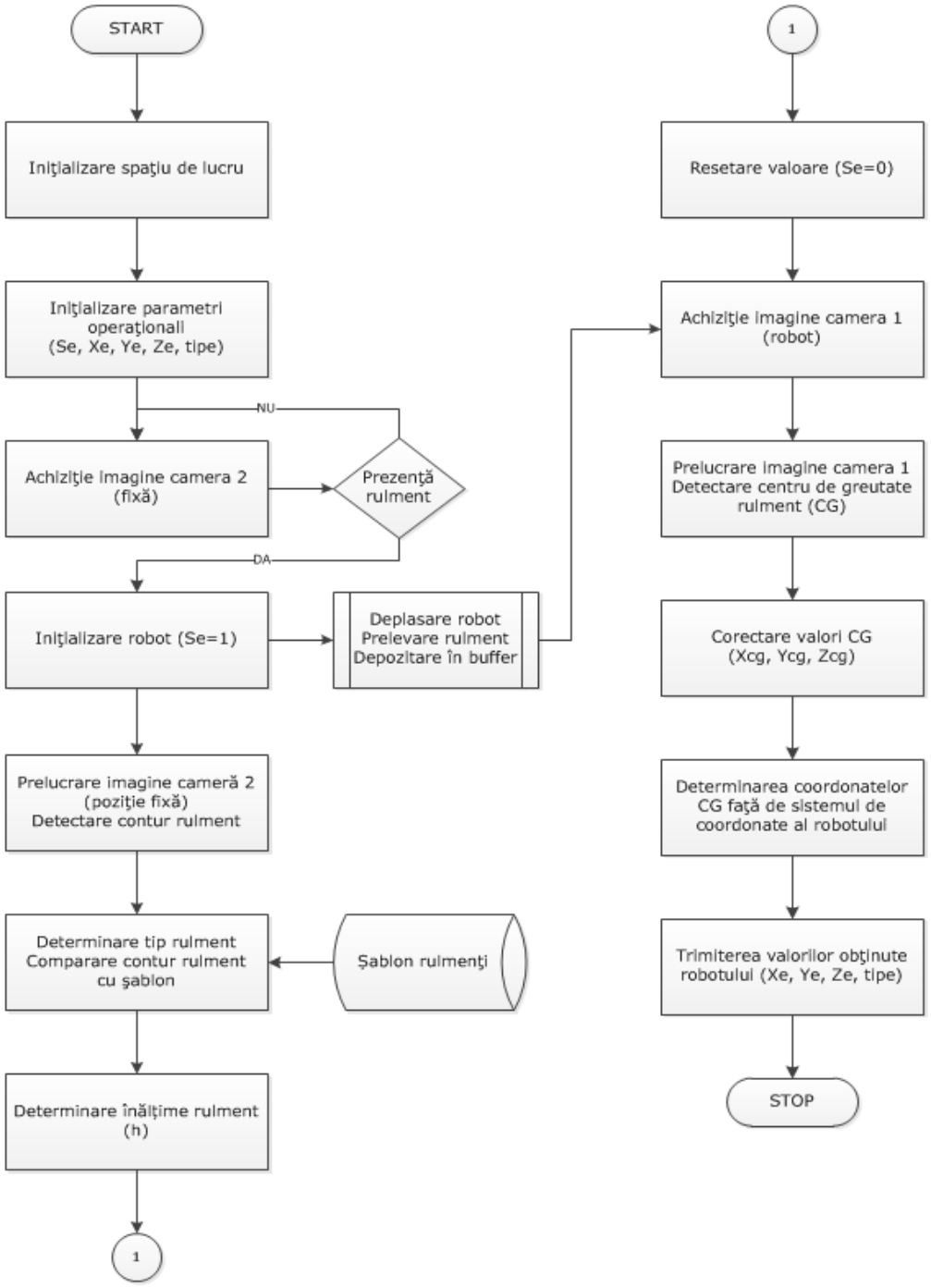


Figura 7. 3: Ordinograma aplicaţiei de determinare a tipului de rulment pe bază de şablon şi a poziţiei acestuia în buffer.

Pasul 9. Pe baza diametrului exterior al rulmentului şi pe baza parametrilor de calibrare ai camerei 1 sunt corectate coordonatele centrului de greutate;

Pasul 10. Informaţiile cu privire la tipul rulmentului şi coordonatele centrului de greutate corectate sunt utilizate pentru comandarea robotului;

Pasul 11. Robotul, pe baza acestor informații, preia rulmentul din buffer şi îl stivuiește în locaşul corespunzător din depozit.

Algoritmul de prelucrare a imaginii care se aplică la pasul 3 este prezentat în figura 7.4. Acest algoritm, ce se foloseşte pentru detectarea obiectului în imagine, se bazează pe faptul că, operaţiunea de segmentare a imaginii se efectuează în spaţiul de culoare HSV, cu scopul de a îmbunătăţi precizia de detectare a obiectelor.

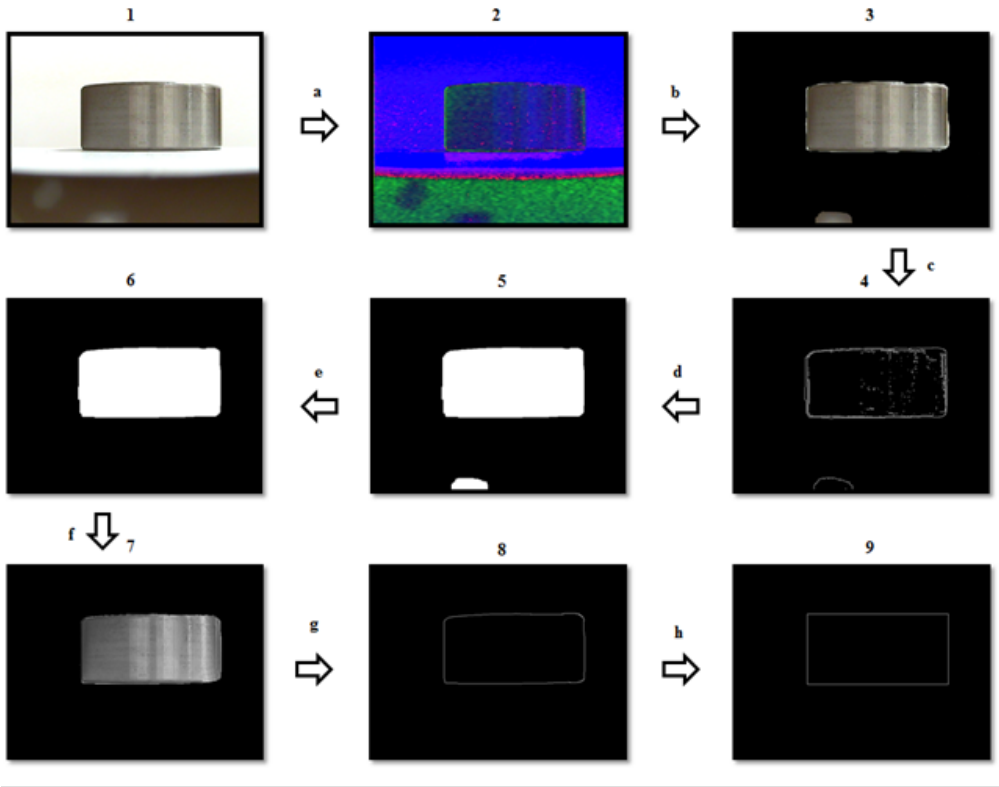


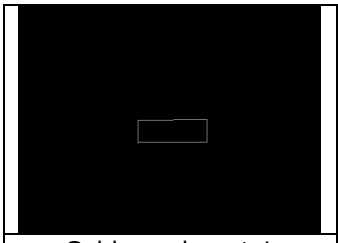
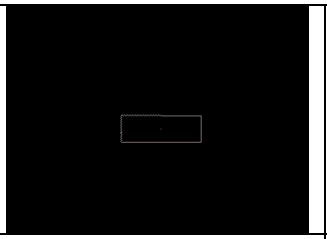
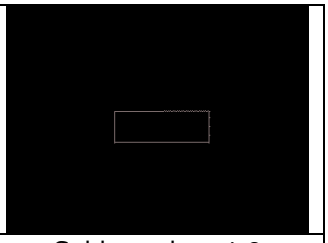
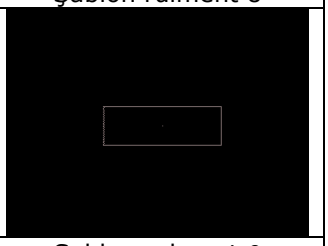
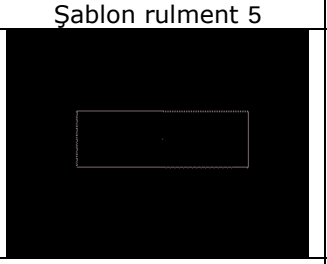
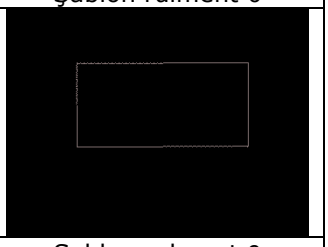
Figura 7. 4: Algoritm de detectare a obiectului vizat: 1) imaginea originală, 2) imaginea HSV, 3) imaginea cu obiectul detectat, 4) imaginea cu muchii, 5) imaginea cu obiectul plin, 6) imagine fără obiecte mici, 7) rulmentul detectat, 8) conturul rulmentului, 9) conturul îndreptat.

Operaţiunile cele mai importante ale acestui algoritm sunt:

a) conversia imagini RGB în imagine HSV;
b) extragerea din imagine a entităţilor incluse în limitele de prag;
c) detectarea marginilor pe baza operatorului Sobel;
d) umplerea contururilor obţinute;
e) eliminare obiectelor mici;
f) extragerea rulmentului din imaginea originală;
g) detectarea conturului rulmentului;
h) corecţia conturului.

Limitele pragului HSV sunt obţinute într-o etapă anterioară de învăţare. Prin intermediul unei interfeţe grafice, un operator uman selectează un număr de pixeli ce aparţin fundalului, din imaginile obţinute de la camera CCD 2. Pe baza valorilor acestor pixeli, limitele minime şi maxime ale pragului sunt calculate.

Tot într-o etapă premergătoare rulării aplicaţiei, şabloanele folosite de cel de-al doilea algoritm de procesare a imaginii (pasul 4) sunt generate(vezi tabelul 7.3). Acestea sunt obţinute prin selectarea celor patru colţuri ale rulmentului, pe imaginile achiziţionate de la camera 2 (vedere din faţă a rulmentului). Aceste modele sunt asociate cu caracteristicile geometrice reale (diametru exterior şi lăţimea) ale rulmenţilor.

		
Şablon rulment 1	Şablon rulment 2	Şablon rulment 3
		
Şablon rulment 4	Şablon rulment 5	Şablon rulment 6
		
Şablon rulment 7	Şablon rulment 8	Şablon rulment 9

Tabelul 7. 3: Tabel cu şabloanele învăţate pentru identificarea rulmenţilor.

Practic cel de-al doilea algoritm de procesare a imaginii (anexa 9) compară, conturul rulmentului generat de primul algoritm cu conturul şabloanelor prin suprapunerea centrelor de greutate. Şablon, care se apropie cel mai mult de conturul detectat, este selectat. Acest lucru este posibil datorită faptului că rulmenţii sosesc întotdeauna pe paletă, în aceeaşi poziţie fixă.

Cel de-al treilea algoritm de procesare de imaginii (pasul 8), pe baza imaginilor cu vederea de sus a rulmenţilor, detectează obiectele rotunde şi centrul lor de greutate raportat la sistemul de referinţă al camerei 1 (figura 7.5). Acest proces se realizează prin aplicarea imaginilor a unui prag global folosind metoda Otsu (Otsu, 1979) şi o condiţie de rotunjime.

Următorul pas (pasul 9), transpune coordonatele (xp, yp) din sistemul de referinţă imaginea în sistemul de referinţă al senzorului video, prin operaţia de

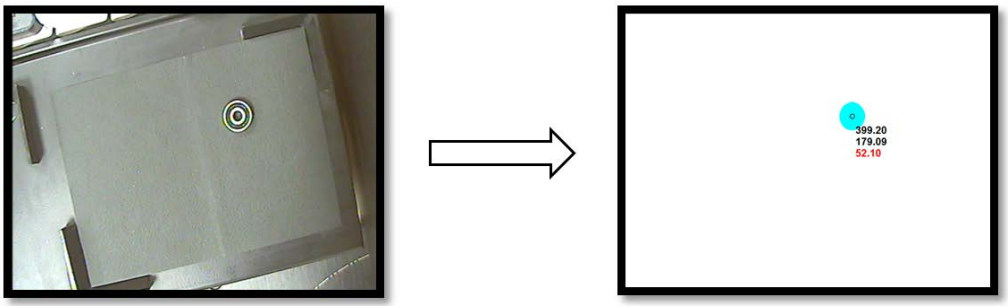


Figura 7. 5: Rezultatul aplicării algoritmului 3 de prelucrare a imaginilor, de la pasul 8.

normalizarea realizată pe baza parametrilor intrinseci ai camerei 1 şi funcţia "normalize.m" (din toolbox-ul de calibrare).

Relaţia dintre coordonatele normalizate α şi β şi coordonatele centrului de greutate ${}^{C1}CG$ (X_G, Y_G, Z_G), în sistemul de referinţă al piesei „P” raportat la camera 1 este:

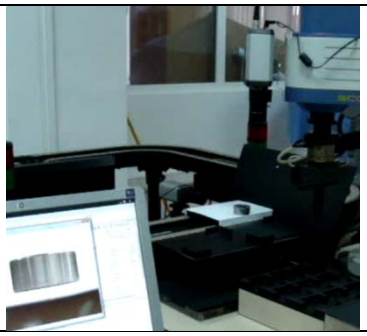
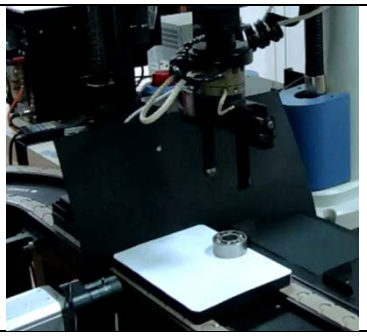
$$\begin{bmatrix} \alpha \\ \beta \end{bmatrix} = \begin{bmatrix} {}^{C1}X_G / {}^{C1}Z_G \\ {}^{C1}Y_G / {}^{C1}Z_G \end{bmatrix} \tag{7.1}$$

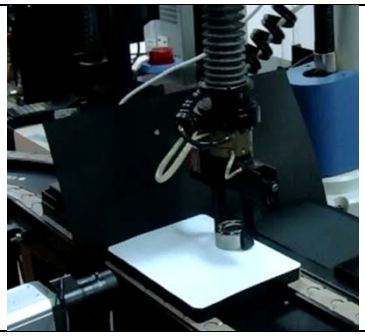
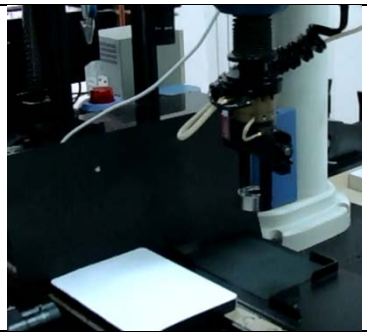
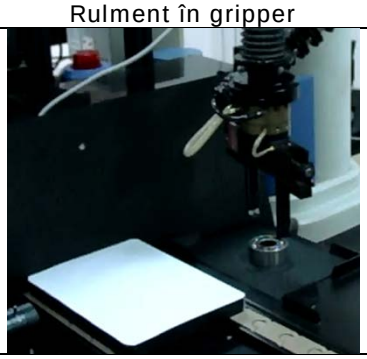
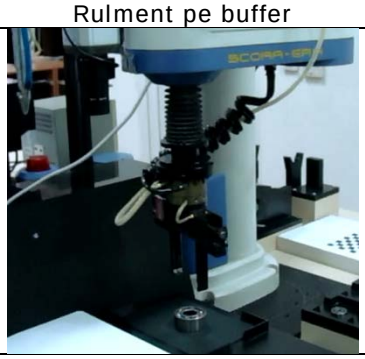
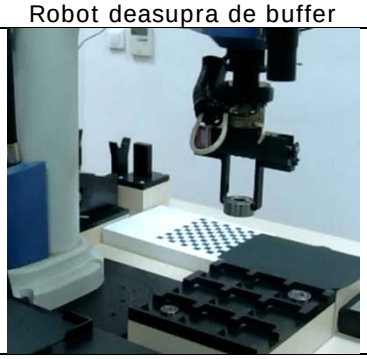
Pentru a se putea obţine coordonatele punctului CG în sistemul de referinţă „P” este necesar ca parametrii extrinseci (tabelul 7.2) şi lăţimea B notată cu h a rulmentului (determinată la pasul 4) să se introducă în ecuaţia. 7.2.

$$\begin{aligned} \alpha * {}^{C1}Z_G &= Rc\_ext_{11} * X_G + Rc\_ext_{12} * Y_G + Tc\_ext_{11} \\ \beta * {}^{C1}Z_G &= Rc\_ext_{21} * X_G + Rc\_ext_{22} * Y_G + Tc\_ext_{21} \\ {}^{C1}Z_G &= Rc\_ext_{31} * X_G + Rc\_ext_{32} * Y_G + Tc\_ext_{31} - h \end{aligned} \tag{7.2}$$

Dacă se rezolvă sistemul, coordonatele centrului de greutate CG, respectiv X_G, Y_G şi Z_G sunt determinate.

Rezultatele obţinute în urma rulării aplicaţiei de vedere computerizată sunt prezentate în tabelul 7.4.

	
Achiziţie imagine rulment pe cărucior	Robot deasupra de cărucior

	
Rulment în gripper	Rulment deasupra de buffer
	
Rulment pe buffer	Achiziţie imagine rulment pe buffer
	
Robot deasupra de buffer	Rulment în gripper
	
Rulment deasupra de depozit	Rulment în locaşul din depozit

Tabelul 7. 4: Rezultatele aplicaţiei de detectare, recunoaştere, manipulare şi depozitar a unui rulment aflat în poziție cunoscută.

7.2. Aplicaţie de detectare şi recunoaştere a tipului de obiect aflat într-o poziţie necunoscută

Scopul aplicaţiei: Detectarea şi recunoaşterea automată a obiectelor ce se regăsesc într-o poziţie necunoscută, în postul de aşteptare (buffer-ul staţiei), pe bază unui algoritm de prelucrări de imagini, pentru sortarea şi stivuirea acestora, în locaşurile corespunzătoare, dintr-un depozit de piese.

7.2.1. Aplicaţia propriu-zisă

Cea de-a doua aplicaţie, se diferenţiază prin faptul că rulmentul nu mai soseşte pe paletă într-o poziţie fixă, el regăsindu-se într-o poziţie necunoscută în buffer. Identificarea tipului de rulment nu se mai realizează în aceiaşi manieră, respectiv algoritmul de comparaţie cu şabloanele nu se mai foloseşte (Pop, 2012b).

Operația de calibrare a camerelor, se reface pentru postul de așteptare, utilizându-se de aceasta dată un șablon de calibrare de tip „tablă de șah” 3D. Modalitatea de calibrare este identică, cu cea utilizată și la aplicaţia anterioară, utilizându-se modelul prezentat în capitolul 6 (subcapitolul 6.2). Acest lucru este posibil deoarece, fiecare camera vizualizează în momentul calibrării, doar un plan al șablonului de calibrare. Şablonul de calibrare 3D este utilizat pentru a se putea realiza o corelaţie între parametri camerei 1 şi parametri camerei 2.

Pentru realizarea acestei aplicaţii de vedere robotizată s-a ţinut cont de următoarele premise:

- Datele de intrare:
  - În postul de aşteptare (buffer) a staţiei de lucru se găseşte în momentul rulării aplicaţiei un singur rulment
  - Situarea rulmentului pe buffer este necunoscută;
  - Senzorii camerelor video sunt paraleli cu planul $O_pX_pY_p$ şi $O_pY_pZ_p$ (proiecţie normală) iar axele optice formează între ele un unghi de aproximativ 90^0;
  - Pragurile necesare algoritmului de detectare a obiectelor, în spaţiul de culoare HSV (Hue, Saturation, Value sau Nuanţă, Saturaţie, Valoare) a imagini prelevată de camera 2 (fixă) sunt cunoscute (se obţin într-o operație anterioare de prelucrare);
- Datele de ieşire:
  - Tipul de rulment aflat în postul de aşteptare, în momentul rulării aplicaţiei;
  - Poziţia rulmentului faţă de sistemul de referinţă ataşat postul de aşteptare (sistemul de referinţă al piesei).

Ordinograma aplicației este ilustrată în figura 7.6. Paşii ce sunt necesari a fi parcurşi pentru realizarea aplicaţiei sunt următorii:

Pasul 1. Rulment este plasat într-o poziţie nedefinită în buffer;

Pasul 2. Calculatorul pe care rulează aplicaţia, comandă robotul SCORA ER 14, pentru ca acesta să se deplaseze dintr-o poziţie de aşteptare într-o poziţie definită de prelevare a imaginii cu ajutorul camere 1;

Pasul 3. Camera CCD 1 şi camera CCD 2 prelevează imaginile obiectului în această poziţie (vederea din faţă şi de sus);

Pasul 4. Primul algoritm detectează conturul rulmentului în imaginea prelevată de camera CCD 1 (un cerc);

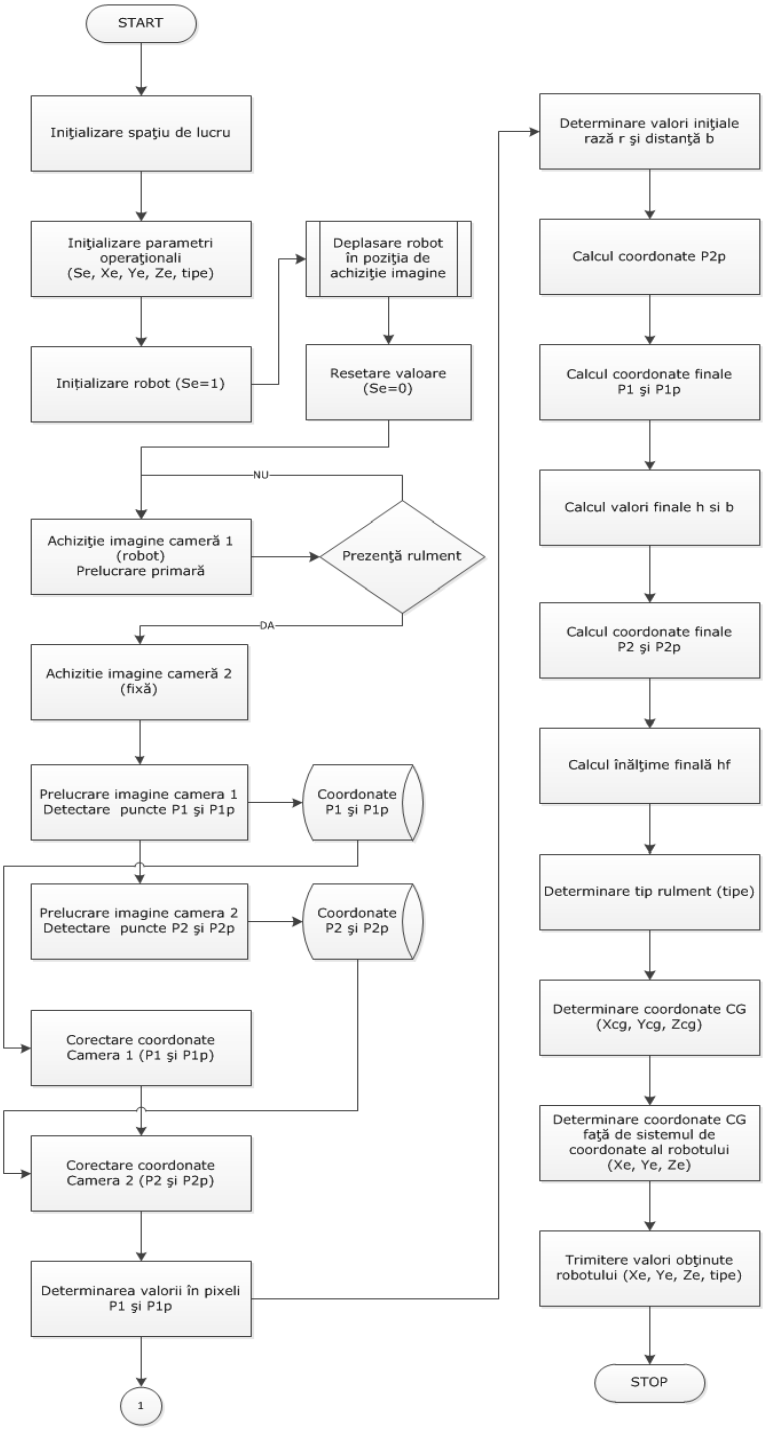


Figura 7. 6: Ordinograma aplicaţiei de determinare a tipului de rulment şi a poziţiei acestuia în buffer pe baza unui model matematic.

Pasul 5. Pe baza acestui contur se determină coordonatele în pixeli (faţă de sistemul de coordonate al imaginii) a doua puncte (P1 şi P3) ce aparţin rulmentului;

Pasul 6. Al doilea algoritm detectează, în imaginea prelevată de camera CCD 2, rulmentul (un dreptunghi);

Pasul 7. Utilizând conturul acestuia se determină coordonatele a altor două puncte ce aparţin rulmentului (P3 şi P4), în pixeli;

Pasul 8. Coordonatele celor patru puncte (figura 7.7) obținute în cele doua imagini sunt normalizate pe baza funcţiei "normalize.m" din toolbox-ul de calibrare.

Pasul 9. Se determină coordonatele în milimetri ale celor patru puncte faţă de sistemul de referinţă ataşat senzorilor vizuali şi se corectează prin eliminarea distorsiunilor;

Pasul 10. Un algoritm matematic, ce realizează corelarea dintre sistemele de referinţă ataşate senzorilor vizuali şi sistemul de referinţă ataşat spaţiului piesei, determină poziția centrului de greutate şi tipul de rulment;

Pasul 11. Informaţiile cu privire la tipul rulmentului şi coordonatele centrului de greutate, sunt utilizate pentru conducerea robotului;

Pasul 12. Robotul, pe baza acestor informații, preia rulmentul din buffer şi îl stivuiește în locaşul corespunzător din depozit;

Pasul 13. Algoritmi de prelucrare de imagini, amintiţi la pasul 4 şi pasul 6, sunt identici cu cei prezentaţi la aplicaţia anterioară (vezi pasul 3 şi pasul 8).

Figura 7. 7: Rezultatele celor doi algoritmi de prelucrare a imaginilor

7.2.2. Modelul matematic de determinare a poziției rulmentului în buffer pe baza a patru puncte.

Modelul matematic utilizat, realizează trecerea din sistemul de coordonate ataşat senzorului vizual al camerei video 1, respectiv al camerei video 2 în sistemul de coordonate ataşat piesei corespunzător camerei 1 şi respectiv camerei 2. Se realizează de asemenea o corelarea a sistemelor de coordonate ale celor două camere video (rezultate în urma calibrării camerelor) cu sistemul de coordonate al piesei. Punctele necesare sunt. (figura 7.8):

- Punctul P1 obţinut din imaginea achiziționata de camera 1 reprezintă centrul de greutate al cercului (rulmentul - vederea de sus);
- Punctul P2 obţinut din imaginea achiziționata de camera 2 reprezintă centrul de greutate al dreptunghiului (rulmentul - vederea frontală);

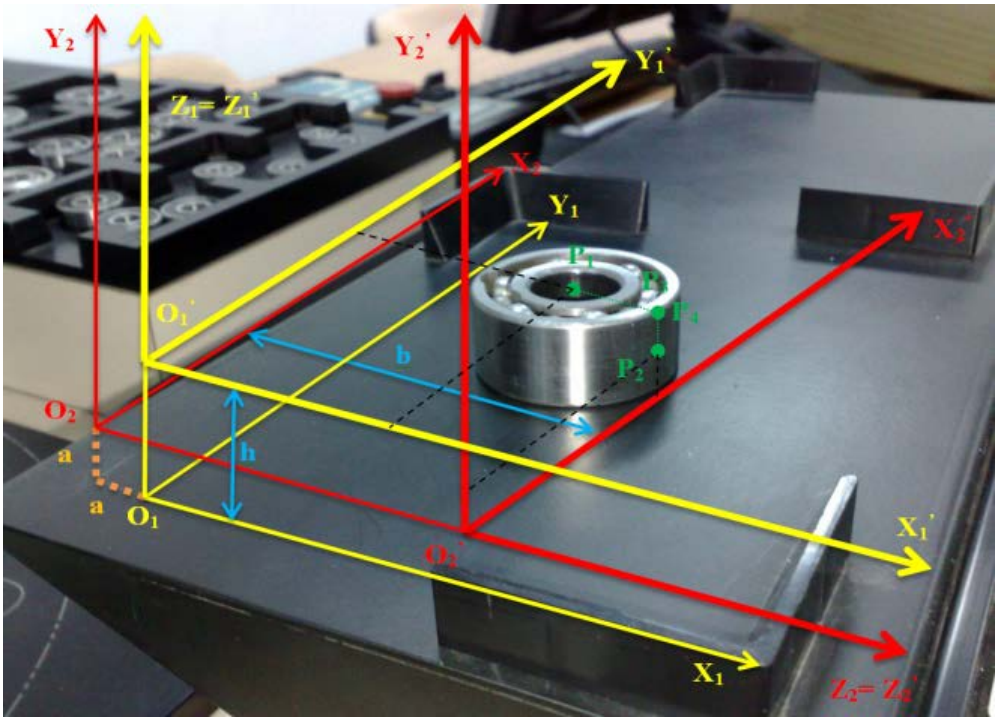


Figura 7. 8: Sistemul de coordonate al buffer-ului: a=distanţa între O1 şi O2; b=deplasarea O2' faţă de O2; h= înălţimea rulmentului (Pop, 2012b).

Punctele P3 (notat P1') şi P4 (notat P2'), care au fost la rândul lor obținute, tot din imaginile achiziţionate de camera 1, respectiv camera 2, reprezintă același punct real. Etapele modelului matematic ce trebuie parcurse sunt următoarele:

Etapa 1. Calcularea coordonatelor inițiale ale punctului P1 în pixeli.

- Se considera configuraţia sistemelor de coordonate din figura 7.9 obţinute în urma calibrării celor doua camere cu şablonul de calibrare 3D.
- Se consideră coordonatele punctului P1 faţă de sistemul de coordonate ca fiind şi coordonatele punctului P2 faţă de sistemul ca fiind .
- Se determina relațiile de legătură între coordonatele punctului P1 şi P2:

$$\begin{cases} {}^1X_{P1} = b - a - r \\ {}^1Y_{P1} = {}^2X_{P2} \\ {}^1Z_{P1} = h \end{cases} \quad \begin{cases} {}^2X_{P2} = {}^1Y_{P1} \\ {}^2Y_{P2} = a - h/2 \\ {}^2Z_{P2} = 0 \end{cases} \tag{7.3}$$

unde a este distanţa între O1 şi O2, b deplasarea O2' faţă de O2 şi h este înălţimea rulmentului.

- Pe baza parametrilor extrinseci obținuți în urma calibrării celor doua camere (matricele de rotaţie şi vectorii de translație) (7.4)

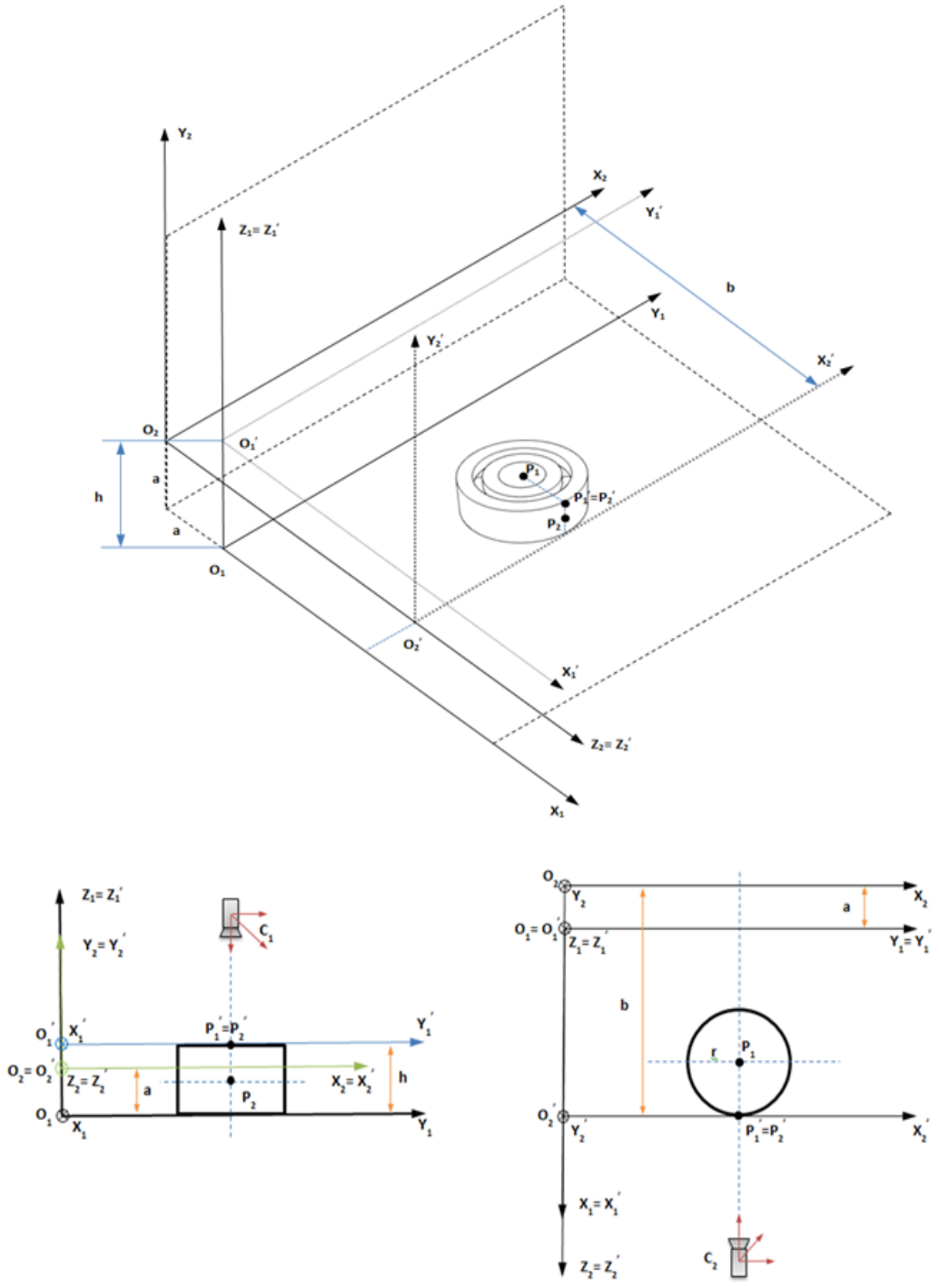


Figura 7. 9: Configuraţia sistemelor de coordonate ale imaginilor în planul piesei

$$\begin{cases} {}^{C1}Rc\_ext = \begin{bmatrix} {}^{C1}Rc\_ext_{11} & {}^{C1}Rc\_ext_{12} & {}^{C1}Rc\_ext_{13} \\ {}^{C1}Rc\_ext_{21} & {}^{C1}Rc\_ext_{22} & {}^{C1}Rc\_ext_{23} \\ {}^{C1}Rc\_ext_{31} & {}^{C1}Rc\_ext_{32} & {}^{C1}Rc\_ext_{33} \end{bmatrix} \\ {}^{C2}Rc\_ext = \begin{bmatrix} {}^{C2}Rc\_ext_{11} & {}^{C2}Rc\_ext_{12} & {}^{C2}Rc\_ext_{13} \\ {}^{C2}Rc\_ext_{21} & {}^{C2}Rc\_ext_{22} & {}^{C2}Rc\_ext_{23} \\ {}^{C2}Rc\_ext_{31} & {}^{C2}Rc\_ext_{32} & {}^{C2}Rc\_ext_{33} \end{bmatrix} \\ {}^{C1}Tc\_ext = \begin{bmatrix} {}^{C1}Tc\_ext_{11} \\ {}^{C1}Tc\_ext_{21} \\ {}^{C1}Tc\_ext_{31} \end{bmatrix} \\ {}^{C2}Tc\_ext = \begin{bmatrix} {}^{C2}Tc\_ext_{11} \\ {}^{C2}Tc\_ext_{21} \\ {}^{C2}Tc\_ext_{31} \end{bmatrix} \end{cases} \quad (7.4)$$

se determină sistemul de ecuaţii:

$$\begin{aligned} \alpha * {}^{C1}Z_{P1} &= {}^{C1}Rc\_ext_{11} * {}^{1}X_{P1} + {}^{C1}Rc\_ext_{12} * {}^{1}Y_{P1} + {}^{C1}Tc\_ext_{11} \\ \beta * {}^{C1}Z_{P1} &= {}^{C1}Rc\_ext_{21} * {}^{1}X_{P1} + {}^{C1}Rc\_ext_{22} * {}^{1}Y_{P1} + {}^{C1}Tc\_ext_{21} \\ {}^{C1}Z_{P1} &= {}^{C1}Rc\_ext_{31} * {}^{1}X_{P1} + {}^{C1}Rc\_ext_{32} * {}^{1}Y_{P1} + {}^{C1}Tc\_ext_{31} - h \end{aligned} \quad (7.5)$$

(unde α şi β reprezintă coordonatele x si y normalizate faţă de camera 1 a punctului P1 în milimetri) şi sistemul:

$$\begin{aligned} \gamma * {}^{C2}Z_{P2} &= {}^{C2}Rc\_ext_{11} * {}^{2}X_{P2} + {}^{C2}Rc\_ext_{12} * {}^{2}Y_{P2} + {}^{C2}Tc\_ext_{11} \\ \delta * {}^{C2}Z_{P2} &= {}^{C2}Rc\_ext_{21} * {}^{2}X_{P2} + {}^{C2}Rc\_ext_{22} * {}^{2}Y_{P2} + {}^{C2}Tc\_ext_{21} \\ {}^{C2}Z_{P2} &= {}^{C2}Rc\_ext_{31} * {}^{2}X_{P2} + {}^{C2}Rc\_ext_{32} * {}^{2}Y_{P2} + {}^{C2}Tc\_ext_{31} - b \end{aligned} \quad (7.6)$$

unde γ şi δ reprezintă coordonatele x si y normalizate faţă de camera 2 a punctului P2.

- Prin prelucrare matematică rezultă un sistem de 4 ecuaţii cu 4 necunoscute. a cărei matrice este:

$$S = \begin{bmatrix} S_{11} & S_{12} & S_{13} & S_{14} \\ S_{21} & S_{22} & S_{23} & S_{24} \\ S_{31} & S_{32} & S_{33} & S_{34} \\ S_{41} & S_{42} & S_{43} & S_{44} \end{bmatrix} \quad (6.7)$$

$$
\begin{aligned}
S_{11} &= {}^{1}X_{P1} * {}^{C1}Rc\_ext_{32} - {}^{C1}Rc\_ext_{12} \\
S_{12} &= {}^{1}X_{P1} * {}^{C1}Rc\_ext_{31} - {}^{C1}Rc\_ext_{11} \\
S_{13} &= -{}^{1}X_{P1} \\
S_{14} &= {}^{1}X_{P1} * {}^{C1}Rc\_ext_{31} - {}^{C1}Rc\_ext_{11} \\
S_{21} &= {}^{1}Y_{P1} * {}^{C1}Rc\_ext_{32} - {}^{C1}Rc\_ext_{22} \\
S_{22} &= {}^{1}Y_{P1} * {}^{C1}Rc\_ext_{31} - {}^{C1}Rc\_ext_{21} \\
S_{23} &= -{}^{1}Y_{P1} \\
S_{24} &= {}^{1}Y_{P1} * {}^{C1}Rc\_ext_{31} - {}^{C1}Rc\_ext_{21} \\
S_{31} &= {}^{2}X_{P2} * {}^{C2}Rc\_ext_{31} - {}^{C2}Rc\_ext_{11} \\
S_{32} &= 0 \\
S_{33} &= 0.5 * {}^{2}X_{P2} * {}^{C2}Rc\_ext_{32} - 0.5 * {}^{C2}Rc\_ext_{12} \\
S_{34} &= -{}^{2}X_{P2} \\
S_{41} &= {}^{2}Y_{P2} * {}^{C2}Rc\_ext_{31} - {}^{C2}Rc\_ext_{21} \\
S_{42} &= 0 \\
S_{43} &= 0.5 * {}^{2}Y_{P2} * {}^{C2}Rc\_ext_{32} - 0.5 * {}^{C2}Rc\_ext_{22} \\
S_{44} &= -{}^{2}Y_{P2}
\end{aligned}
\tag{7.8}
$$

unde determinantul sistemului S este:

$$ds = \det(S) \tag{7.9}$$

– Vectorul soluțiilor se exprimă ca:

$$D = \begin{bmatrix} d_{11} \\ d_{21} \\ d_{31} \\ d_{41} \end{bmatrix}$$

$$
\begin{aligned}
d_{11} &= {}^{C1}Tc\_ext_{11} - {}^{1}X_{P1} * {}^{C1}Tc\_ext_{31} + {}^{1}X_{P1} * {}^{C1}Rc\_ext_{31} * a - {}^{C1}Rc\_ext_{11} * a \\
d_{21} &= {}^{C1}Tc\_ext_{21} - {}^{1}Y_{P1} * {}^{C1}Tc\_ext_{31} + {}^{1}Y_{P1} * {}^{C1}Rc\_ext_{31} * a - {}^{C1}Rc\_ext_{21} * a \\
d_{31} &= {}^{C2}Tc\_ext_{11} + a\left({}^{2}X_{P2} * {}^{C2}Rc\_ext_{32} - {}^{C2}Rc\_ext_{12}\right) - {}^{2}X_{P2} * {}^{C2}Tc\_ext_{31} \\
d_{41} &= {}^{C2}Tc\_ext_{21} + a\left({}^{2}Y_{P2} * {}^{C2}Rc\_ext_{32} - {}^{C2}Rc\_ext_{22}\right) - {}^{2}Y_{P2} * {}^{C2}Tc\_ext_{31}
\end{aligned}
\tag{7.10}
$$

– Se determină prima necunoscută a sistemului (${}^{1}Y_{P1}$) notată cu y:

$$my = \begin{bmatrix} d_{11} & S_{12} & S_{13} & S_{14} \\ d_{21} & S_{22} & S_{23} & S_{24} \\ d_{31} & S_{32} & S_{33} & S_{34} \\ d_{41} & S_{42} & S_{43} & S_{44} \end{bmatrix}$$

$$dy = \det(my) \tag{7.11}$$

$$y = \frac{dy}{ds}$$

– Se determină raza iniţială a rulmentului r:

$$mr = \begin{bmatrix} S_{11} & d_{12} & S_{13} & S_{14} \\ S_{21} & d_{22} & S_{23} & S_{24} \\ S_{31} & d_{32} & S_{33} & S_{34} \\ S_{41} & d_{42} & S_{43} & S_{44} \end{bmatrix}$$

$$dr = \det(mr) \tag{7.12}$$

$$r = \frac{dr}{ds}$$

– Se determină înălțimea inițială a rulmentului:

$$mh = \begin{bmatrix} S_{11} & S_{12} & d_{13} & S_{14} \\ S_{21} & S_{22} & d_{23} & S_{24} \\ S_{31} & S_{32} & d_{33} & S_{34} \\ S_{41} & S_{42} & d_{43} & S_{44} \end{bmatrix}$$

$$dh = \det(mh) \tag{7.13}$$

$$h = \frac{dh}{ds}$$

– Se determină distanţa b (distanţa dintre O_2 şi O_2')

$$mb = \begin{bmatrix} S_{11} & S_{12} & S_{13} & d_{14} \\ S_{21} & S_{22} & S_{23} & d_{24} \\ S_{31} & S_{32} & S_{33} & d_{34} \\ S_{41} & S_{42} & S_{43} & d_{44} \end{bmatrix}$$

$$db = \det(mb) \tag{7.14}$$

$$b = \frac{db}{ds}$$

– Odată determinate cele patru necunoscute, se calculează cu relația (7.3) coordonatele punctului P1.

Etapa 2. În mod similar se calculează coordonatele inițiale ale punctului P3 în pixeli.

Etapa 3. Calcularea valorii razei r şi dimensiunii b, pe baza punctelor P1 şi P3;

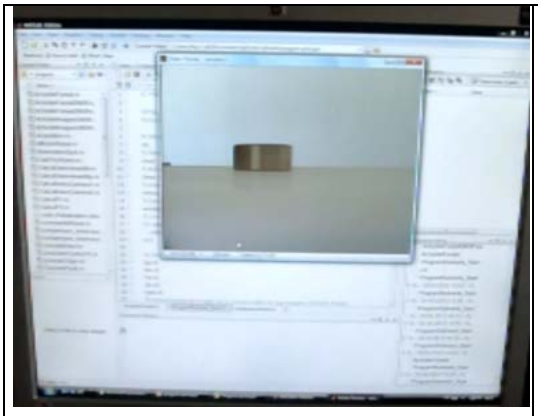
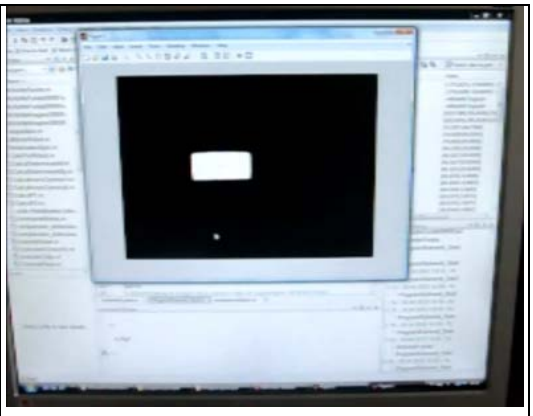
Etapa 4. Determinarea coordonatelor punctului P4 (X_{P4}, Y_{P4}) şi a distanţei *h* rescriind corespunzător ecuația (7.6);

Etapa 5. Următoarea parte a algoritmului, conţine o buclă iterativă, care permite determinarea cu o precizie mai mare a coordonatelor calculate. Începând cu valoarea inițială b, se determină noile coordonate ale punctelor P1 şi P3 şi valorile lui b şi h.

Etapa 6. Obţinerea coordonatelor finale ale punctelor P1, P2, P3, P4 şi valorile lui b, h şi r.

Etapa 7. Determinarea coordonatelor centrului de greutate al rulmentului pe baza rezultatelor obţinute

Rezultatele obţinute în urma rulării aplicației de vedere computerizată sunt prezentate în tabelul 7.5.

	
Achiziţie imagine	Prelucrare imagine
	
Comandare robot prelevare piesă	Depozitare piesă

Tabelul 7. 5: Rezultatele aplicaţiei de detectare, recunoaştere, manipulare şi depozitarea unui rulment aflat în poziție necunoscută.

7.3. Aplicaţie de detectare, recunoaştere, identificare, sortare şi manipulare a unor obiecte suprapuse

Scopul aplicaţiei: Detectarea şi recunoaşterea automată a mai multor obiecte ce se regăsesc într-o poziţie necunoscută, în postul de aşteptare (buffer-ul staţiei), pe bază unui algoritm de prelucrări de imagini, pentru sortarea şi stivuirea acestora, în locaşurile corespunzătoare, dintr-un depozit de piese.

Aplicaţia trei, se diferenţiază de celelalte două, prin faptul că, în postul de aşteptare al staţiei de lucru, se afla în acelaşi timp, mai mulţi rulmenţi a căror poziţie şi orientare este necunoscută. În mod asemănător, ca la aplicaţia anterioara (aplicaţia doi), identificarea rulmentului se realizează pe baza caracteristicilor geometrice calculate, nu cu ajutorul şabloanelor învăţate într-o etapa anterioară (cum se realizează la prima aplicaţie).

Deosebirea faţă de aplicaţia doi, care conferă acestei variante un grad de complexitate mai ridicat, constă în faptul că coordonatele punctelor P2 şi P4, puncte necesare soluţionării modelului matematic (vezi secţiunea 7.2.2), ce permite determinarea tipului şi situării rulmentului, nu pot fi extrase din imagine în acelaşi mod. În acest caz, sunt necesare operaţii de prelucrare a imaginii suplimentare, deoarece în imaginea 2 (prelevată de camera video fixă) apar mai mulţi rulmenţi ce se suprapun.

O altă diferenţă, faţa de cele două aplicaţii anterioare, este dată de faptul că în acest caz nu se mai utilizează algoritmul de segmentare a imaginii în spaţiul HSV. Detectarea pieselor se realizează prin extragerea fundalului din imaginile achiziţionate pe parcursul rulării aplicaţiei.

Matricea de transformare şi parametri intrinseci şi extrinseci ai camerelor video obţinuţi în urma operaţiei de calibrare a robotului şi a camerelor, realizată pentru aplicaţia doi, se reutilizează şi pentru această aplicaţie. Acest lucru este posibil, întrucât rulmenţi se situează tot în postul de aşteptare al staţiei de lucru, iar poziţia şi orientarea camerelor video în raport cu sistemul de referinţă ataşat spaţiului piesei nu s-a modificat.

Standul experimental şi echipamentele utilizate sunt cele ce au fost utilizate şi în cadrul aplicaţiei doi.

Pentru realizarea acestei aplicaţii de vedere robotizată s-a ţinut cont de următoarele premise:

- Datele de intrare:
  - În postul de aşteptare a staţiei de lucru se găseşte în momentul rulării aplicaţiei mai mulţi rulmenţi
  - Situarea rulmenţilor pe buffer este necunoscută;
  - Senzorii camerelor video sunt paraleli cu planul $O_pX_pY_p$ şi $O_pY_pZ_p$ (proiecţie normală) iar axele optice formează între ele un unghi de aproximativ 90^0;
  - Imaginile cu postul de aşteptare gol, respectiv imaginile cu fundalul spaţiului de lucru, sunt achiziţionate înaintea rulării aplicaţiei, când în buffer nu se află piese.
- Datele de ieşire:
  - Tipul rulmenţilor ce se găsesc în postul de aşteptare, în momentul rulării aplicaţiei;
  - Poziţia rulmenţilor faţă de sistemul de referinţă ataşat postul de aşteptare (sistemul de referinţă al piesei).

Ordinograma aplicaţiei este ilustrată în figura 7.10 şi 7.11 (continuare).

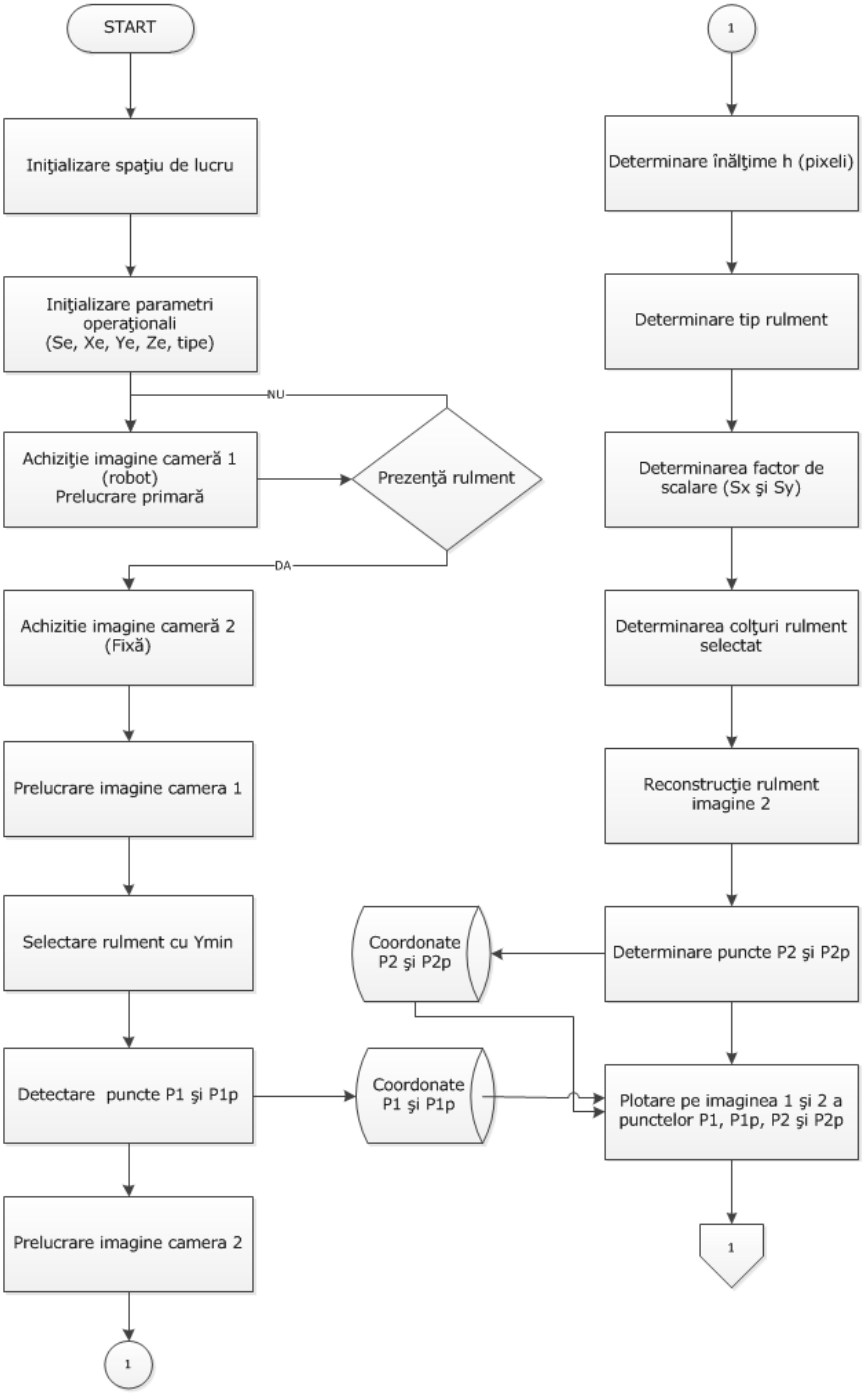


Figura 7. 10: Ordinograma aplicaţiei de detectare, recunoaștere, sortare manipulare şi depozitare a unor rulmenți suprapuși situaţi în buffer-ul stației de lucru (partea 1).

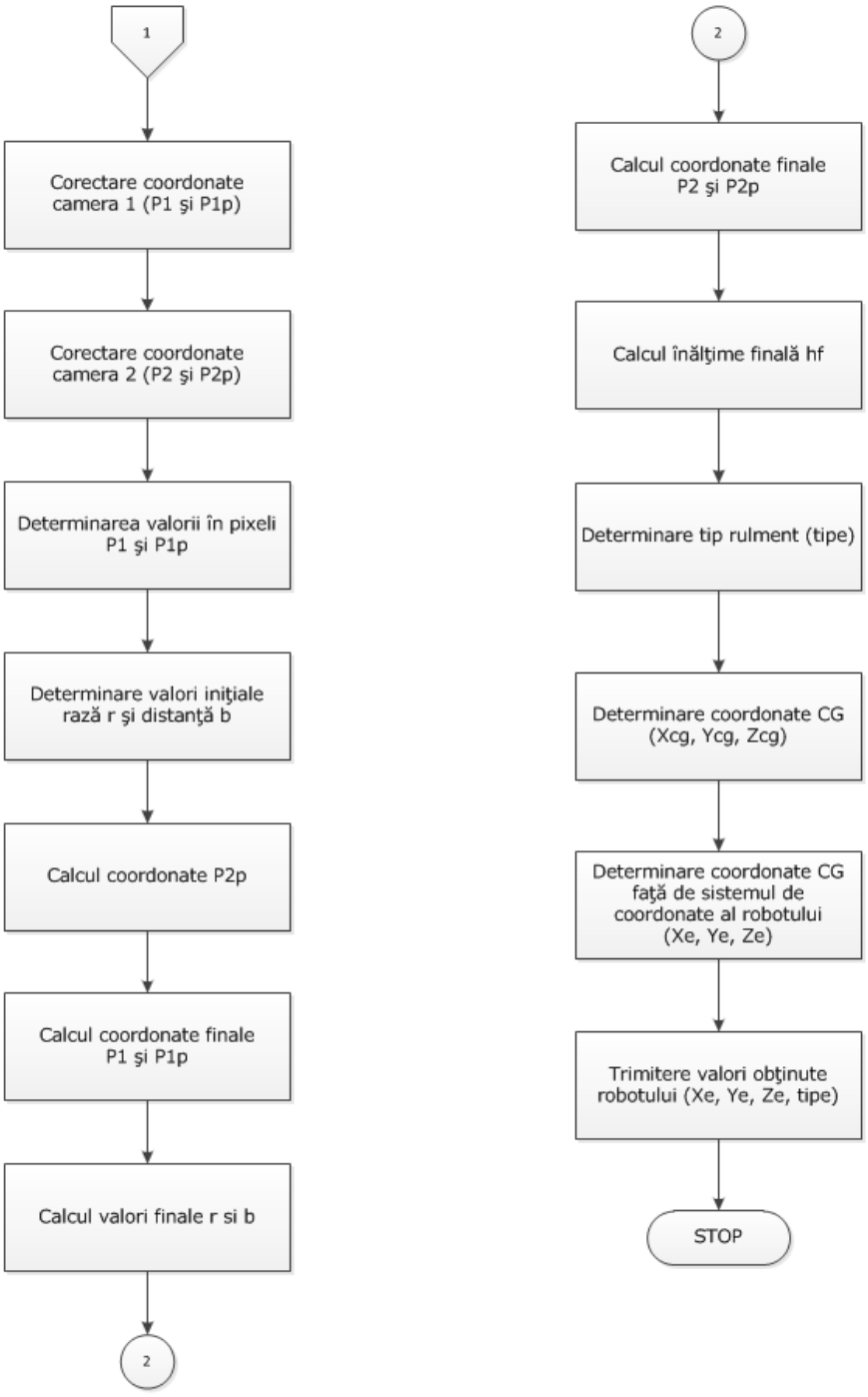


Figura 7. 11: Ordinograma aplicaţiei de detectare, recunoaştere, sortare manipulare şi depozitare a unor rulmenți suprapuși situaţi în buffer-ul stației de lucru (partea 2).

Paşii, ce sunt necesari a fi parcurşi, pentru realizarea aplicaţiei sunt următorii:

Pasul 1. Calculatorul pe care rulează aplicaţia, comandă robotului SCORA ER 14, să se deplaseze dintr-o poziţie de aşteptare într-o poziţie definită de achiziţie a imaginii cu ajutorul camere 1;

Pasul 2. Camera CCD 1 şi camera CCD 2 achiziţionează imagini cu postul de aşteptare gol (imagini cu fundalul spaţiului de lucru). Comanda de execuţie a achiziţiei acestor imagini, se face l-a începutul rulării algoritmului, de fiecare dată când buffer-ul este gol, pentru a evita erorile de prelucrare care pot apărea din cauza unor factori externi;

Pasul 3. Aplicaţia este prevăzută cu un timp de aşteptare, timp în care rulmenţii sunt plasaţi în poziţii aleatoare, nedefinite, în buffer-ul staţiei de lucru;

Pasul 4. După scurgerea timpului de aşteptare, camera video 1 şi 2 prelevează imagini cu rulmenţii aflaţi în buffer;

Pasul 5. Dacă în buffer sunt prezenţi rulmenţi, algoritmul execută pasul următor. În caz contrar execută pasul 4.

Pasul 6. Primul algoritm de prelucrare de imagini, execută operaţia de extragere a fundalului 1 din imaginea 1 şi detectează rulmenţii prezenţi;

Figura 7. 12: Detectarea rulmenţilor în imaginea 1

Pasul 7. Un al doilea algoritm de prelucrare de imagini rectifică conturul rulmenţilor detectaţi. Se determină raza medie (în pixeli) a fiecărui rulment şi centrul de masă. Pe baza acestor informaţii conturul rulmenţilor este reconstruit. Astfel se reduc erorile de determinare a poziţiei;

Pasul 8. Se determină rulmentul care se află la distanţa minimă de cameră şi este vizualizat integral. Rulmentul în cauză este cel al cărui y în pixeli (faţă de sistemul de coordonate al imaginii 1) este minim. Se determină de asemenea şi un parametru de poziţie în imagine (este mai aproape de marginea din stânga sau de marginea din dreapta).

Pasul 9. Se determină coordonatele iniţiale (în pixeli) ale punctului P1 si P3;

Figura 7. 13: Extragerea rulmentului cu y minim şi determinarea punctelor P1 şi P3

Pasul 10. Un al treilea algoritm de prelucrare de imagini, execută operaţia de extragere a fundalului 2 din imaginea 2 şi detectează rulmenţii prezenţi;

Pasul 11. Alt algoritm, determina pe baza unei corespondente de pixeli între cele doua imagini, înălțimea rulmentului (în pixeli) considerat a fi cel mai aproape de camera 2.

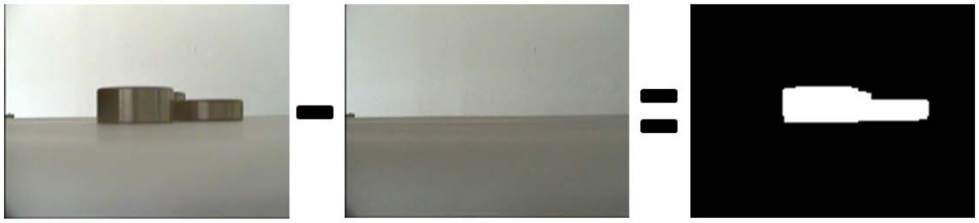


Figura 7. 14: Detectarea rulmenţilor în imaginea 2.

Pasul 12. Pe baza razei şi a înălțimii, în pixeli, se determină tipul de rulment;

Pasul 13. Se calculează factorul de scalare (Sx şi Sy) în imaginea 2;

Pasul 14. Se caută în imaginea 2 (în funcţie de parametrul de poziţie de la pasul 8), una din muchiile rulmentului vizat. Se extrag extremele muchiei (coordonatele acestora), ce reprezintă două colţuri ale rulmentului, şi se calculează pe baza factorilor de scalare celelalte două.

Pasul 15. Un algoritm de reconstrucție generează conturul rulmentului;

Pasul 16. Se extrag coordonatele punctelor P2 şi P4.

Figura 7. 15: Extragerea rulmentului şi determinarea punctelor P2 şi P4

Pasul 17. Se plotează, pe imaginile rezultate (cu rulmentul extras), cele patru puncte;

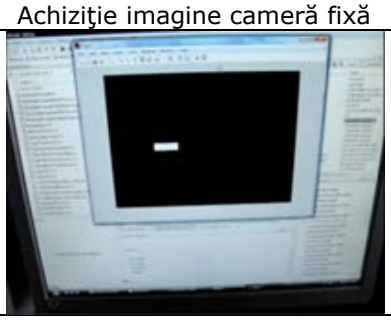
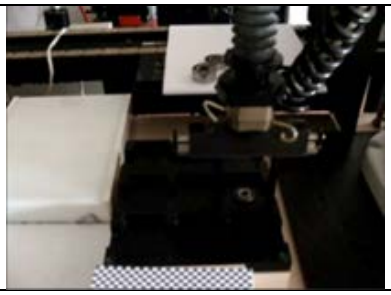
Pasul 18. Coordonatele celor patru puncte obținute în cele doua imagini sunt normalizate pe baza funcţiei "normalize.m" din toolbox-ul de calibrare. Se corectează coordonatele în milimetri a celor patru puncte faţă de sistemul de referinţă ataşat senzorilor vizuali prin eliminarea distorsiunilor.

Pasul 19. Se determină poziţia rulmentului extras, faţă de sistemul de referinţă atașat spaţiului piesei, pe baza modelului matematic prezentat în secţiunea 7.2.2.

Pasul 20. Informaţiile cu privire la tipul rulmentului şi poziţia rulmentului, sunt utilizate pentru comanda robotului;

Pasul 21. Pe baza informaţiilor primite, robotul execută operaţiile de preluare a rulmentul din buffer şi de stivuire a acestuia în locaşul corespunzător din depozit;

Acest proces se repetă, până în momentul în care, în postul de aşteptare a staţiei de lucru nu se mai găseşte nici un rulment.

	
Achiziţie imagine cameră robot	Achiziţie imagine cameră fixă
	
Prelucrare imagine cameră robot	Prelucrare imagine cameră fixă
	
Comandare robot prelevare piesă	Comandare robot depozitare piesă

Tabelul 7. 6: Rezultatele aplicaţiei de detectare, recunoaştere, manipulare şi depozitare a mai multor rulmenţi aflaţi în poziţii necunoscute.

7.4. Concluzii

În urma rulării celor trei aplicaţii s-a observat că prima abordare este mai precisă în localizarea piesei aflate în spaţiul de lucru, dar celelalte două abordari sunt mult mai aproape de condiţiile reale ale unui mediu de lucru nedefinit, deoarece rulmentul sau rulmenţii se aflau în poziţii necunoscute. Cea de-a treia aplicaţie prezintă un grad de complexitate mai ridicat faţă de primele două, datorită faptului că în zona de lucru se regăsesc mai multe piese, cee ce conduce la necesitatea existenţei unui sistem de vedere robotizat mai „inteligent”, capabil să ia anumite decizi pe baza unor informaţii obţinute din imagini.

Utilizarea unor informaţii redundante, obţinute de la ambele camere, conduce la o mai bună fiabilitate a sistemului. În ciuda faptului că forma geometrică a piesei este una simplă, metodele dezvoltate în a doua şi a treia aplicaţie ar putea fi extinse la obiecte mai complexe, bazat pe informaţii CAD 3D.

8. APLICAŢIE DE RECUNOAŞTERE ŞI PREHENSARE A OBIECTELOR

8.1. Introducere

În cadrul acestui capitol, este prezentată o aplicaţie de vedere robotizată, ce se bazează pe un algoritm, dezvoltat în mediul de lucru Matlab, ce permite determinarea punctelor de prehensare a unor obiecte necunoscute, de natură 2D (obiecte 3D, de tip placă, la care ce-a de-a treia dimensiune, este constantă, în comparaţie cu celelalte două)

Scopul aplicaţiei: Detectarea unor piese de tip placă, ce sosesc pe căruciorul unui conveior de transfer, la staţia de lucru robotizată, în vederea determinării punctelor de prehensare, optime, necesare pentru realizarea unei manipulări stabile, de către robot, a acestora.

Aplicaţia presupune şi recunoaşterea pieselor, pe baza unor descriptori globali, pentru realizarea operaţiei de inserare, în locaşurile corespunzătoare, ale unei contrapiese.

Echipamentele utilizate pentru realizarea acestei aplicaţii sunt:

- Un sistem robotizat de manipulare şi asamblare ce are în componenţă următoarele:
  - Robot de tip scara (SCORA ER 14);
  - Controler central şi controler robot (tip B);
  - Calculator central şi calculator de staţie;
  - Cutie de conexiuni;
  - Teach pendant.
- Conveior de transfer ce presupune:
  - Patru cărucioare;
  - Trei posturi de aşteptare;
  - Un PLC.
- Sistem de vedere artificial compus din:
  - O cameră web, montată într-o poziţie fixă, deasupra conveiorului de transfer, la o staţie de lucru.
  - Un calculator (pentru dezvoltarea şi rularea algoritmului de vedere computerizată);
  - O sursa de lumină (o masă de iluminat);
  - Programul de bază Matlab (mediul de lucru în care s-a dezvoltat aplicaţia).
- Un şablon de calibrare 2D de tip „tablă de şah";
- Patru piese cu forme diferite;
- O contrapiesa, respectiv o placă prevăzută cu patru locaşuri pentru cele patru piese.

Placa, şi cele patru piese, utilizate în cadrul acestei aplicaţii sunt prezentate în figura 8.1 şi în figura 8.2 . Atât piesele cât şi contrapiesa sunt realizate din plexiglas. Prelucrarea lor s-a făcut prin tăiere cu jet de apă, jet a cărui grosime a fost setată la 0.5 mm.

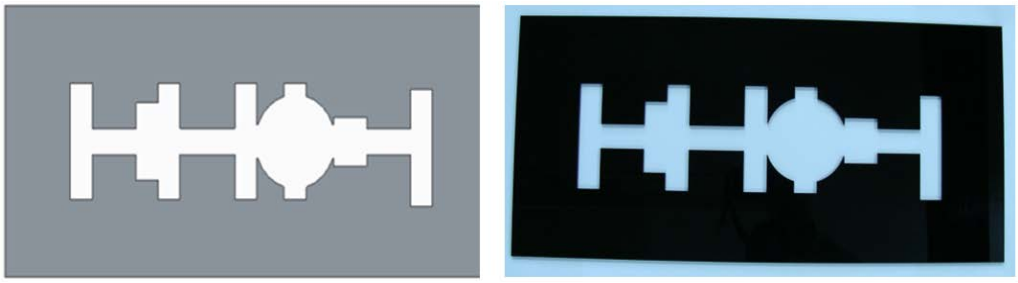


Figura 8. 1: Contrapiesă (varianta CAD şi reală).

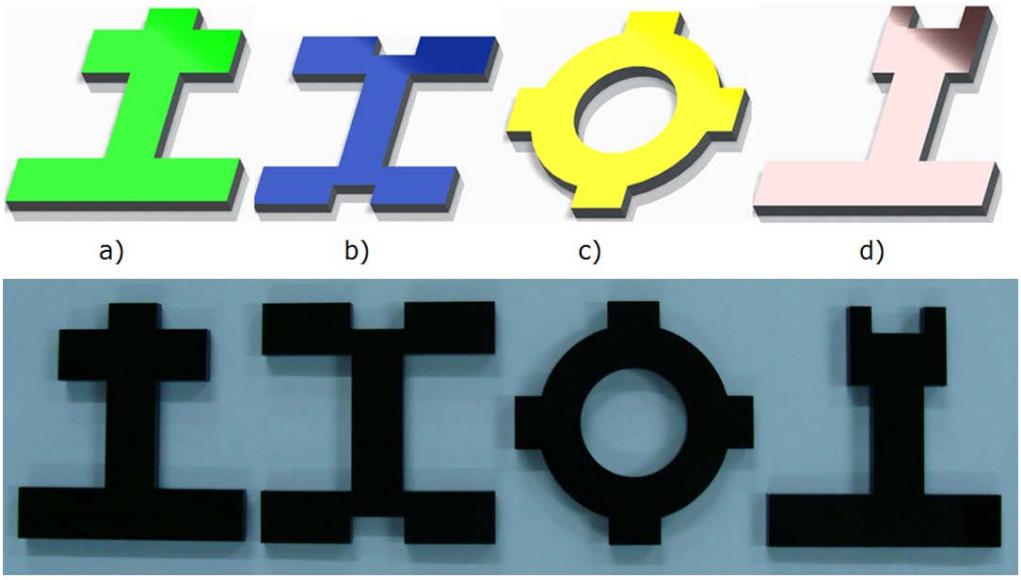


Figura 8. 2: Piesele 2D (reale şi în CAD).

Algoritmul ce realizează determinarea punctelor de prehensare optimă, a unor piese necunoscute, presupune implementarea unei metode bazată pe proprietăţile de simetrie ale acestora.

8.2. Metodă de prehensare bazată pe simetrie

Simetria s-a dovedit că este importantă pentru a caracteriza formele plane în vederea prehensiunii (Solomon, 2011, Pratt, 2001, Symon, 1971, Sanz, 2005, Blake, 1995). În mod obișnuit prinderea se efectuează aplicând o forță de-a lungul axei principale de simetrie sau perpendicular pe ea.

Axele de simetrie pot fi determinate știind că (Symon, 1971):

- orice plan de simetrie al unui corp este perpendicular pe o axă principală de inerție.
- orice axă de simetrie a unui corp este o axă principală de simetrie.

Sanz (Sanz, 2005) utilizează momentele statice pentru a determina candidate pentru punctele de prindere, apoi utilizează o combinație simetrie-curbură pentru a selecta punctele ce satisfac toate condițiile.

Lee *et al.* (Lee, 2006) propune o abordare mai complexă a detectării simetriei, ce utilizează gradienții unei imagini și potrivirea pixelilor muchiilor pentru a afla linii de simetrie, bazat pe transformata Hough.

O metodă hibridă de detectare a axelor de simetrie în imagini binare a fost propusă de Costantini și Casali (Costantini, 2007), bazată atât pe prelucrarea numerică convențională, cât și pe procesarea analogică neliniară cu rețele neuronale.

O abordare în domeniul frecvenței pentru detectarea simetriei este prezentată de Tzimiropoulos *et al.* (Tzimiropoulos, 2008). Reprezentarea polară este utilizată pentru a determina ordinul simetriei.

Deoarece căutarea axelor de simetrie poate produce rezultate multiple, este necesară introducerea unei măsuri a simetriei.

O'Mara și Owens găsesc axele de simetrie pe baza faptului că dacă un obiect binar are simetrie bilaterală, atunci fiecare hiperplan de simetrie bilaterală trebuie să treacă prin centrul de masă al obiectului și să conțină o axă principală de inerție.

Un algoritm simplu de determinare a tuturor axelor de simetrie al obiectelor din imagini monocrome este prezentat de Marola (Marola, 1989). Este utilizată o măsură specială a simetriei. O metrică alternativă a fost definită de Koulkarni *e al.* (Koulkarni, 1995).

Pentru o imagine bidimensională, conținând un singur obiect, dacă $\boldsymbol{P} = (x,y)$ este un punct din imagine, cu intensitatea $B(x,y)$, momentul de ordinul $(p+q)$ al imaginii poate fi calculat cu relația (Solomon, 2011, Pratt, 2001, Symon, 1971, Sonka, 2008):

$$m_{pq} = \sum_{x=1}^{J}\sum_{y=1}^{K} x^p y^q B(x,y) \tag{8.1}$$

în care (J, K) este dimensiunea imaginii. Dacă imaginea este binară (alb/negru), atunci momentele codează direct informația despre formă (Solomon, 2011).

Momentul central de ordinul $(p+q)$ al unei forme bidimensionale poate fi definit ca (Solomon, 2011, Sonka, 2008):

$$\mu_{pq} = \sum_{x=1}^{J}\sum_{y=1}^{K} (x-\overline{x})^p (y-\overline{y})^q B(x,y) \tag{8.2}$$

în care $\bar{x}$ și $\bar{y}$ sunt coordonatele centrului de masă al formei:

$$\begin{aligned} \overline{x} &= \frac{m_{10}}{m_{00}} \\ \overline{y} &= \frac{m_{01}}{m_{00}} \end{aligned} \tag{8.3}$$

Pentru o imagine binară conținând un obiect cu n puncte (x_i, y_i), centrul de masă este definit de valorile medii:

$$\begin{aligned} \overline{x} &= \frac{1}{n}\sum_{i=1}^{n} x_i \\ \overline{y} &= \frac{1}{n}\sum_{i=1}^{n} y_i \end{aligned} \tag{8.4}$$

Introducând varianţele:

$$\begin{aligned} \sigma_x^2 &= \frac{1}{n}\sum_{i=1}^{n}\left(x_i - \overline{x}\right)^2 \\ \sigma_y^2 &= \frac{1}{n}\sum_{i=1}^{n}\left(y_i - \overline{y}\right)^2, \end{aligned} \tag{8.5}$$

şi covarianţele:

$$p_{xy} = \frac{1}{n}\sum_{i=1}^{n}\left(x_i - \overline{x}\right)\left(y_i - \overline{y}\right) \tag{8.6}$$

matricea covarianţei are forma:

$$C = \begin{bmatrix} c_{11} & c_{12} \\ c_{21} & c_{22} \end{bmatrix} = \begin{bmatrix} \sigma_x^2 & p_{xy} \\ p_{xy} & \sigma_y^2 \end{bmatrix} = \frac{1}{n}\begin{bmatrix} \mu_{20} & \mu_{11} \\ \mu_{11} & \mu_{02} \end{bmatrix} \tag{8.7}$$

Momentele de inerţie sunt legate de varianţe şi covarianţe (Pratt, 2001, Papoulis, 1991, McCartin, 2008):

$$I_{xx} = n\sigma_y^2,\ I_{yy} = n\sigma_x^2,\ I_{xy} = -np_{xy} \tag{8.8}$$

astfel că matricea inerţiei poate fi calculată din matricea covarianţei:

$$I = \begin{bmatrix} I_{xx} & I_{xy} \\ I_{xy} & I_{yy} \end{bmatrix} = n\left[\mathrm{tr}(C)\cdot E - C\right] = \begin{bmatrix} \mu_{02} & -\mu_{11} \\ -\mu_{11} & \mu_{20} \end{bmatrix} \tag{8.9}$$

în care tr() este funcţia urma matricei, iar *E* este matricea unitate 2x2. Vectorii proprii ai matricelor de inerţie şi covarianţei sunt paraleli (McCartin, 2008) şi sunt axele principale de inerţie ale formei (Symon, 1971, Goldstein, 2002).

Cunoscând că:

- orice axă de simetrie a unui corp este o axă principală;
- vectorii proprii ai matricei de inerţie sunt axele principale de inerţie ale obiectului;
- vectorii proprii ai matricei de inerţie sunt paraleli cu cei ai matricei covarianţei.

Candidatele pentru axele de simetrie pot fi calculate ca vectori proprii ai matricei covarianței.

Efectuând descompunerea valorilor singulare pentru matricea covarianței conduce la matricea diagonală (Symon, 1971):

$$\mathrm{V}^{\mathrm{T}} \cdot \mathrm{C} \cdot \mathrm{V} = \Lambda \tag{8.10}$$

în care coloanele matricei

$$V = \begin{bmatrix} v_{11} & v_{12} \\ v_{21} & v_{22} \end{bmatrix} \tag{8.11}$$

sunt vectorii proprii ai matricei covarianței *C* și

$$\Lambda = \begin{bmatrix} \lambda_1 & 0 \\ 0 & \lambda_2 \end{bmatrix} \tag{8.12}$$

conține valorile proprii ale matricei *C*. Pentru o imagine bidimensională, valorile pot fi determinate explicit din ecuația valorilor proprii (8.10):

$$\lambda_1 = \frac{1}{2}\left(c_{11} + c_{22} - \sqrt{\left(c_{11} - c_{22}\right)^2 + 4c_{12}c_{21}} \right) \tag{8.13}$$

$$\lambda_2 = \frac{1}{2}\left(c_{11} + c_{22} + \sqrt{\left(c_{11} - c_{22}\right)^2 + 4c_{12}c_{21}} \right) \tag{8.14}$$

$$\begin{aligned} \frac{v_{11}}{v_{21}} &= \frac{\lambda_1 - c_{22}}{c_{21}} \\ \frac{v_{12}}{v_{22}} &= \frac{\lambda_2 - c_{22}}{c_{21}} \end{aligned} \tag{8.15}$$

Din aceste relații se poate determina direcția axei principale de inerție (corespunzătoare inerției minime):

$$\theta = \arctan\left(\frac{\lambda_1 - c_{22}}{c_{12}} \right) \tag{8.16}$$

Notând cu $v_1 = v_{11}/v_{21}$ și $v_2 = v_{12}/v_{22}$ pantele axelor principale și știind că ele sunt ortogonale, adică $v_1 \cdot v_2 = -1$, iar din (15):

$$v_1 + v_2 = \frac{v_1^2 - 1}{v_1} = \frac{c_{11} - c_{22}}{c_{21}} \tag{8.17}$$

se poate obține o altă expresie pentru unghiul θ:

$$\theta = -\frac{1}{2}\arctan\left(\frac{2c_{21}}{c_{11}-c_{22}}\right) = -\frac{1}{2}\arctan\left(\frac{2\mu_{11}}{\mu_{20}-\mu_{02}}\right) \tag{8.18}$$

Dacă $\mu_{20} = \mu_{02}$ ($\lambda_1 = c_{22}$), sau μ_{11} = 0 (c_{12} = 0), unghiul poate fi calculat utilizând momentele de ordin superior (Martin, 1996) sau:

$$\sin 2\theta = \frac{-\mu_{11}}{\sqrt{\mu_{11}^2 + (\mu_{20}-\mu_{02})}} \tag{8.19}$$

$$\cos 2\theta = \frac{\mu_{20}-\mu_{02}}{\sqrt{\mu_{11}^2 + (\mu_{20}-\mu_{02})}} \tag{8.20}$$

Alegerea punctelor de prindere este, în general, un proces decizional complex, necesitând compromisuri între mai multe criterii de evaluare a calității prinderii, bazat pe cunoștințe despre sarcina de rezolvat prin prindere și starea curentă a mediului (Montana, 1992).

Punctele de prindere sunt, uzual, punctele de intersecție ale conturului formei cu una din axele de simetrie sau cu o axă perpendiculară pe una din axele de simetrie și trecând prin centrul de masă.

Selectarea perechii optime de puncte de prindere este bazată pe calcularea unei măsuri a simetriei formei după ce aceasta a fost rotită astfel încât axa de simetrie să devină orizontală (paralelă cu axa Ox), similar cu cele propuse în (Goldstein, 2002, Martin, 1996).

Axa reală de simetrie este considerată a fi dreapta care maximizează măsura simetriei. Punctele de intersecție ale acestei axe cu linia de contur a formei pot fi utilizate ca puncte de prindere.

O soluție mai practică este selectarea drept puncte de prindere a intersecțiilor conturului cu axa de inerție maximă (Sanz, 2005, Ramnath, 2004. Pentru cele mai multe forme întâlnite în practică, această axă este perpendiculară pe axa principală de inerție (de obicei axa de inerție minimă) și trece prin centrul de masă.

Dacă calitatea prinderii este măsurată prin abilitatea de a echilibra forţele din planul de prindere, atunci punctele de prindere sunt selectate astfel încât lungimea coardei definită de ele să fie maximizată (Mirtich, 1994). Acest lucru plasează punctele de prindere pe o formă simetrică la intersecția conturului cu axa principală de simetrie și face să fie îndeplinite condițiile de închidere pentru formă și pentru forțe la prinderea cu două degete (Blake, 1995, Chen, 1993, Nguyen, 1988).

Metoda propusă a fost implementată în Matlab de (Savii, Davidescu, 2012), sub forma unui executabil, autorul tezei utilizând-o ca atare (fără a-i aduce modificări) în aplicaţia de vedere robotizată. Acest exeutabil este apelat în momentul rulării aplicaţiei, iar rezultatele obținute sunt prezentate în continuare.

Cele patru piese, din figura 8.2, proiectate pentru a ilustra robusteţea algoritmului menţionat anterior se caracterizează prin:

- Piesa 1 (figura figura 8.2 a) prezintă o singură axă de simetrie;
- Piesa 2 (figura figura 8.2 b) prezintă două axe de simetrie;
- Piesa 3 (figura figura 8.2 c) prezintă patru axe de simetrie;
- Piesa 4 (figura figura 8.2 d) este asimetrică.

Standul experimental utilizat este prezentat în figura 8.3.

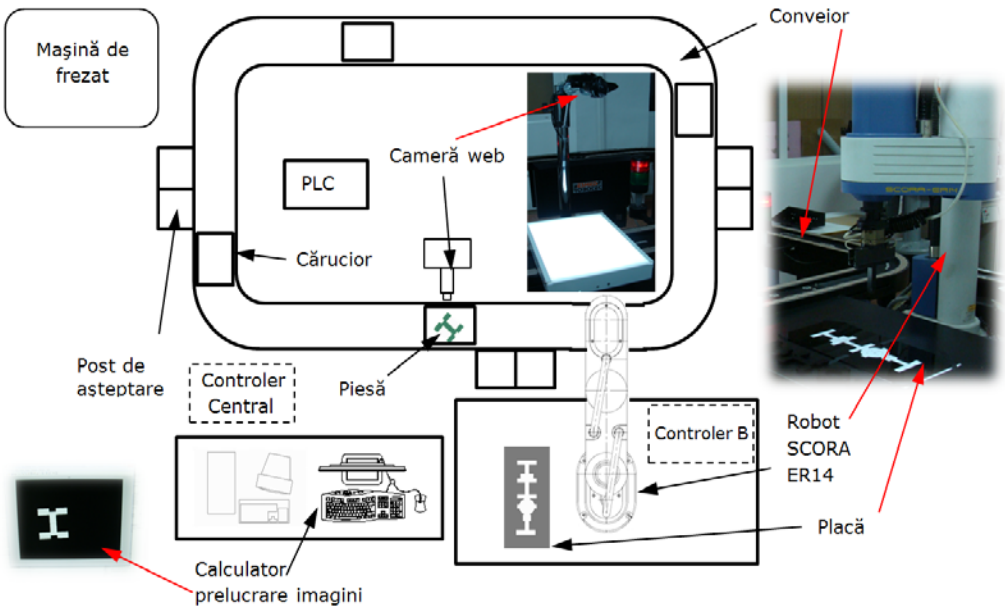


Figura 8. 3: Standul experimental.

8.3. Algoritmul de vedere robotizată

Etapele premergătoare realizării acestei aplicaţii sunt calibrarea robotului SCORA (vezi subcapitolul 6.1.1) şi calibrarea camerei web Philips SPZ 3000 (vezi subcapitolul 6.1.2). Rezultatele obţinute în urma calibrării camerei sunt prezentate în tabelul 8.1 şi figura 8.4.

Parametri intrinseci	**Valori**
Distanţa focală [pixel]:	fc = [1741.82778 1742.47018]
Punctul principal [pixel]:	cc = [663.52299 455.21604]
Unghiul oblic [grade]:	alpha_c = [0.00087]
Distorsiunile:	kc = [--0.17140 1.65057 0.00325 0.00117 -5.51531]
Erori de pixeli[pixel]:	err = [0.29422 0.27879]
Parametri extrinseci	**Valori**
Vectorul de translaţie [mm]:	Tc_ext =[-143.06988 -65.30095 493.360269]
Vectorul de rotaţie [rad]:	omc_ext = [-2.201401 -2.239195 -0.027376]
Matricea de rotaţie [rad]:	Rc_ext = [-0.017095 0.999791 0.011236 0.999767 0.016944 0.013403 0.013210 0.011463 -0.999847]
Erori de pixeli [pixel]:	err = [0.26383 0.21595]

Tabelul 8. 1: Parametri camerei web Philips SPZ 3000.

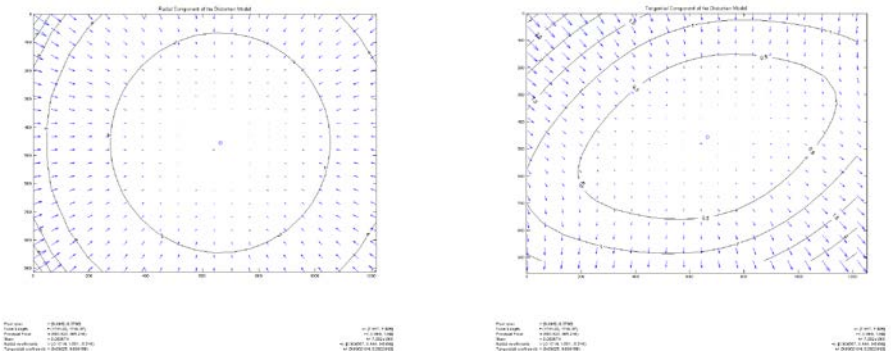


Figura 8. 4: Distorsiunile radiale şi tangenţiale ale lentilei camerei web.

Pentru realizarea acestei aplicaţii, s-a ţinut cont de următoarele premise:

- Datele de intrare:
  - Se utilizează un singur cărucior al conveior-ului de transfer, ce este prevăzut cu o masă de iluminat. Acest lucru s-a realizat pentru a simplifica algoritmul de prelucrare a imaginii, prin reducerea artefactelor introduse de iluminarea neuniformă.
  - Pe cărucior soseşte la staţia de lucru o singură piesă;
  - Piesa nu se cunoaşte;
  - Situarea piesei pe cărucior nu este cunoscută;
  - Senzorul camerei web este paralel cu planul piesei (proiecţie normală);
  - O imagine cu căruciorul prevăzut cu masa de iluminat, goală, este achiziţionată înaintea rulării aplicaţiei.
  - Contrapiesa este fixată pe o masă, în spaţiul de lucru al robotului;
  - Poziţiile centrelor de greutate a locaşurilor, corespunzătoare celor patru piese, din contrapiesă, sunt învăţate (figura 8.5).

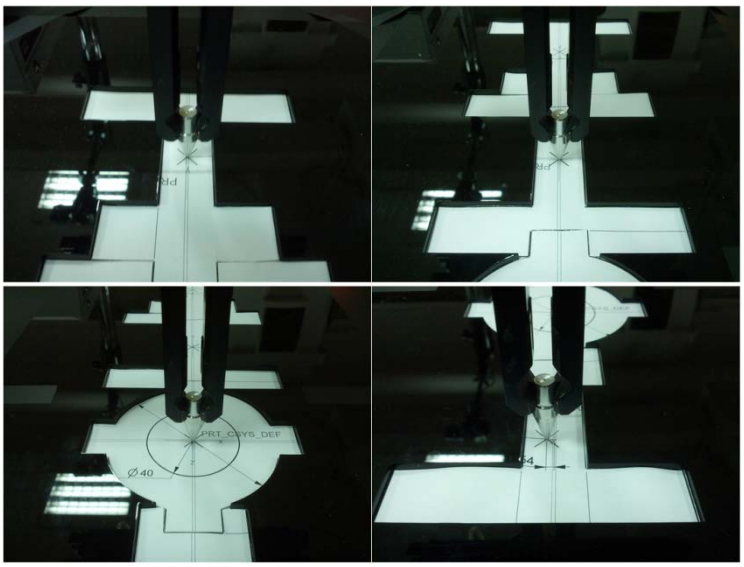


Figura 8. 5: Poziţiile centrelor de greutate ale locaşurilor pieselor din contrapiesă.

- Datele de ieşire:
  - Tipul piesei aflate pe cărucior în momentul rulării aplicaţiei;
  - Situarea piesei (poziţia şi orientarea piesei) pe cărucior.

Ordinograma aplicaţiei este ilustrată în figura 8.6.

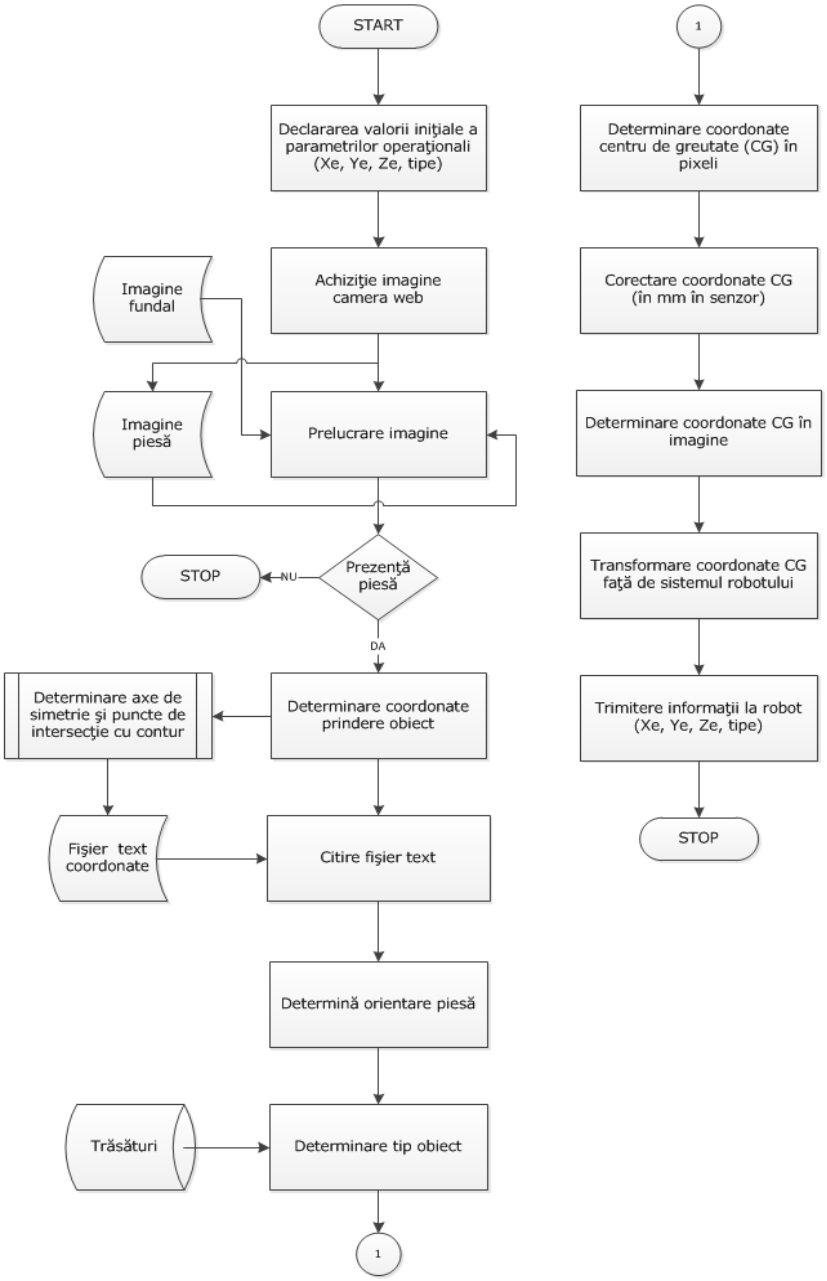


Figura 8. 6: Ordinograma aplicaţiei de determinare a punctelor de prehensare.

Paşii ce sunt necesari a fi parcurşi pentru realizarea aplicaţiei sunt următorii:

Pasul 1. O piesă necunoscută este plasată pe masa de iluminat, la staţia de frezat, într-o poziţie oarecare;

Pasul 2. Calculatorul central comandă PLC-ul pentru eliberarea căruciorului pentru a se putea deplasa până la următoarea staţie de lucru;

Pasul 3. Piesa soseşte la staţia de asamblare, unde este oprită în dreptul camerei video;

Pasul 4. Se porneşte programul de vedere robotizată ce permite camerei web să achiziţioneze o imagine, cu piesa necunoscută pe masa de iluminat.

Pasul 5. Un algoritm de prelucrare de imagini extrage fundalul achiziţionat într-o etapă anterioară şi determină prezenţa sau absenţa piesei pe masă (figura 8.7);

Figura 8. 7: Detectare obiect necunoscut.

Pasul 6. În cazul în care nu există piesă pe masă sau algoritmul nu detectează nimic, aplicaţia se opreşte. În caz contrar, se execută următorul pas;

Pasul 7. Se apelează algoritmul de determinare a punctelor optime de prehensare. Etapele acestui algoritm sunt:

a. Algoritmul determină toate axele de simetrie ale piesei (dacă există). În cazul în care piesa nu are axe de simetrie, se determină dreapta care conferă valoarea maximă a coeficientului de simetrie;
b. Selectează pe baza unor condiţii impuse maxim două axe principale;
c. Determină punctele care apar la intersecţia axelor cu conturul piesei;
d. Salvează coordonatele acestor puncte într-un fişier text (cu extensie .txt).

Rezultatele obţinute pentru cele patru piese, în urma parcurgerii acestui pas, sunt prezentate în figura 8.8.

Pasul 8. Se citeşte fişierul text şi se aleg, pe baza unei condiţii de dimensiune (distanţa dintre bacurile dispozitivului de prehensare), coordonatele unei perechi de puncte. Punctele respective definesc o axă de simetrie, ce formează cu orizontala, ce trece prin centrul de greutate al piesei (în imagine), două unghiuri (în cadranul 1 şi 4). Se alege dintre cele două, unghiul minim pe baza căruia se determină unghiul de rotaţie al axei 4 a robotului (roll-ul).

Pasul 9. Un alt algoritm, determină pe baza imaginii rezultate şi pe baza unor descriptori globali tipul piesei (figura 8.9, 8.10, 8.11).

Pasul 10. Se determină, în pixeli, faţă de sistemul de coordonate al imaginii, coordonatele centrului de greutate (CG) al piesei.

Pasul 11. Aceste coordonate sunt corectate prin eliminarea erorilor introduse de distorsiunile lentilelor.

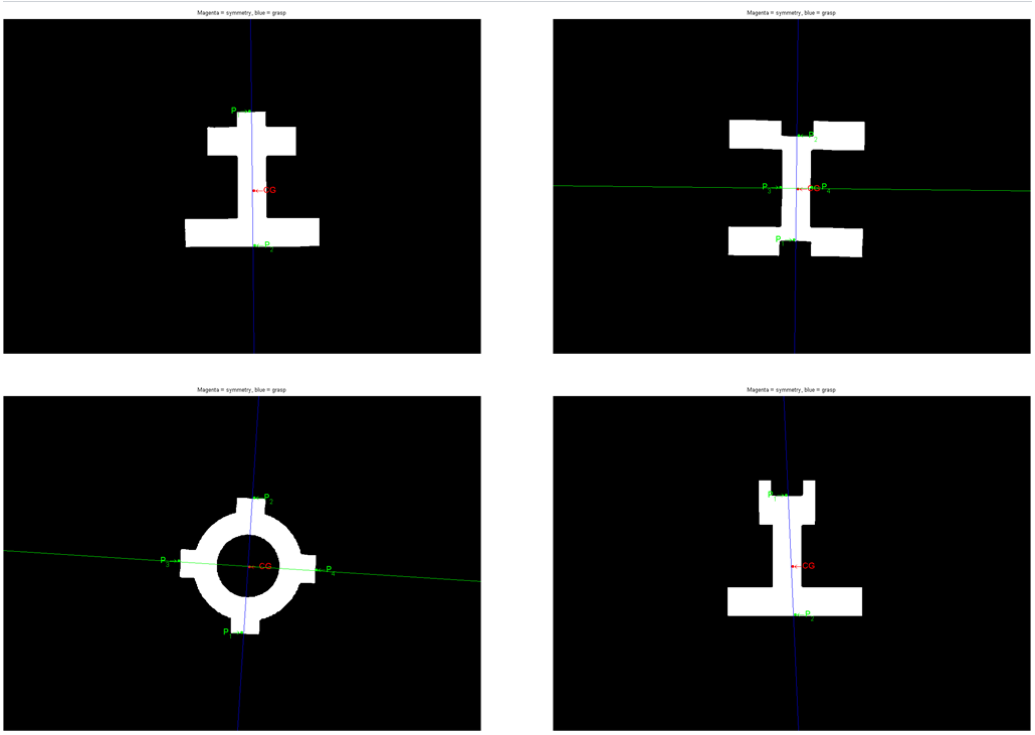


Figura 8. 8: Axele de simetrie şi punctele de prehensare determinate pentru cele patru piese.

Pasul 12. Informaţiile cu privire la tipul piesei şi coordonatele centrului de greutate sunt utilizate pentru comanda robotului. În același timp, se comandă conveiorului de transfer, pentru a-i permite căruciorului să se deplaseze la staţia de lucru a robotului.

Pasul 13. Robotul, pe baza acestor informaţii, preia piesa de pe masa de iluminat şi o inserează în contrapiesă (anexa 10).

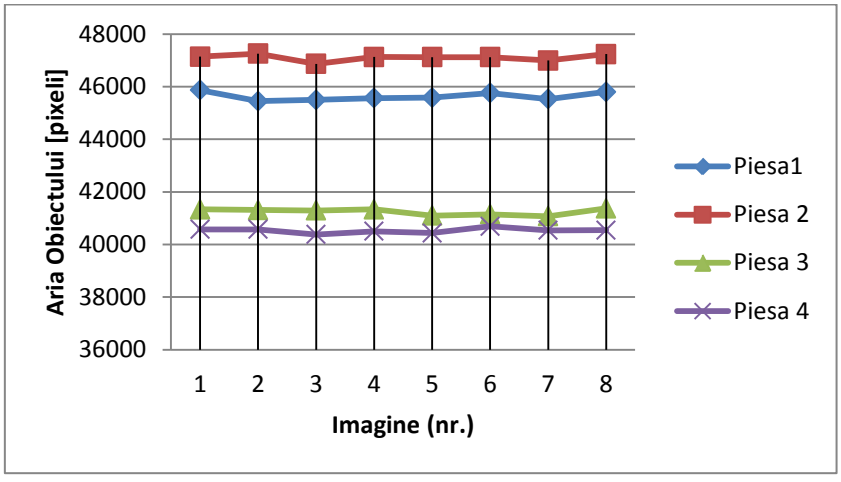


Figura 8. 9: Aria obiectelor în pixeli.

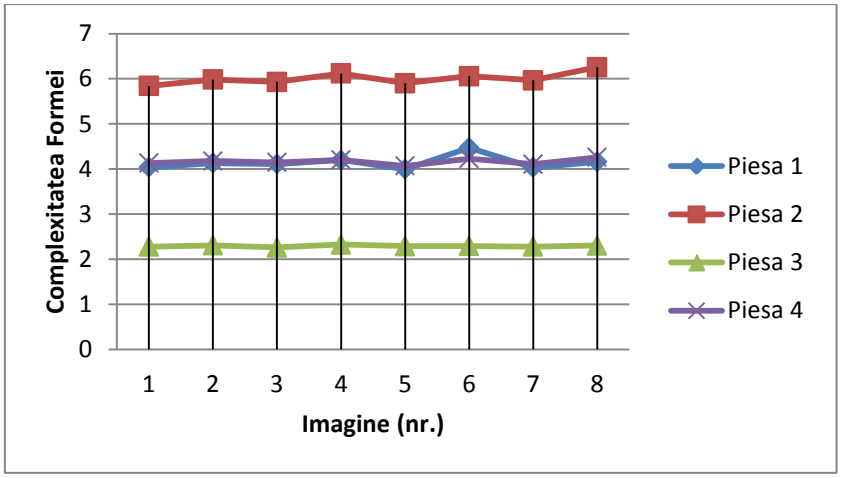


Figura 8. 10: Complexitatea formei obiectelor.

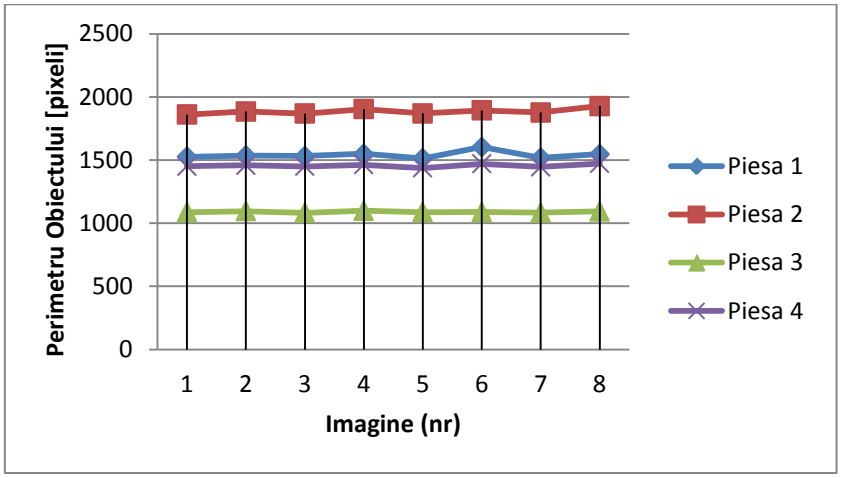


Figura 8. 11: Perimetrul obiectelor în pixeli.

8.4. Rezultate şi concluzii

În practică, programarea roboților este foarte importantă din punct de vedere al prelevării corecte de către robot a piesei, astfel încât la închiderea şi deschiderea dispozitivului de prehensare piesa sa nu se mişte, respectiv sa nu îşi modifice poziţia şi orientarea.

Atingerea acestui deziderat, la programarea on-line, înseamnă timp îndelungat de manevrare a dispozitivului în poziția necesară.

Utilizarea unui program de determinare a simetriei pieselor poate să rezolve această problemă în mod automat.

Operația de programarea off-line, respectiv realizarea unui program pe calculator care să includă modelul virtual al robotului şi modelul spațiu/masa de lucru, poate conduce la obținerea unor rezultate favorabile dar a căror reproducere în realitate să fie dificilă.

Prin urmare, se poate trage concluzia că orice program off-line realizat, este necesar a fi testat inițial printr-o verificare online.

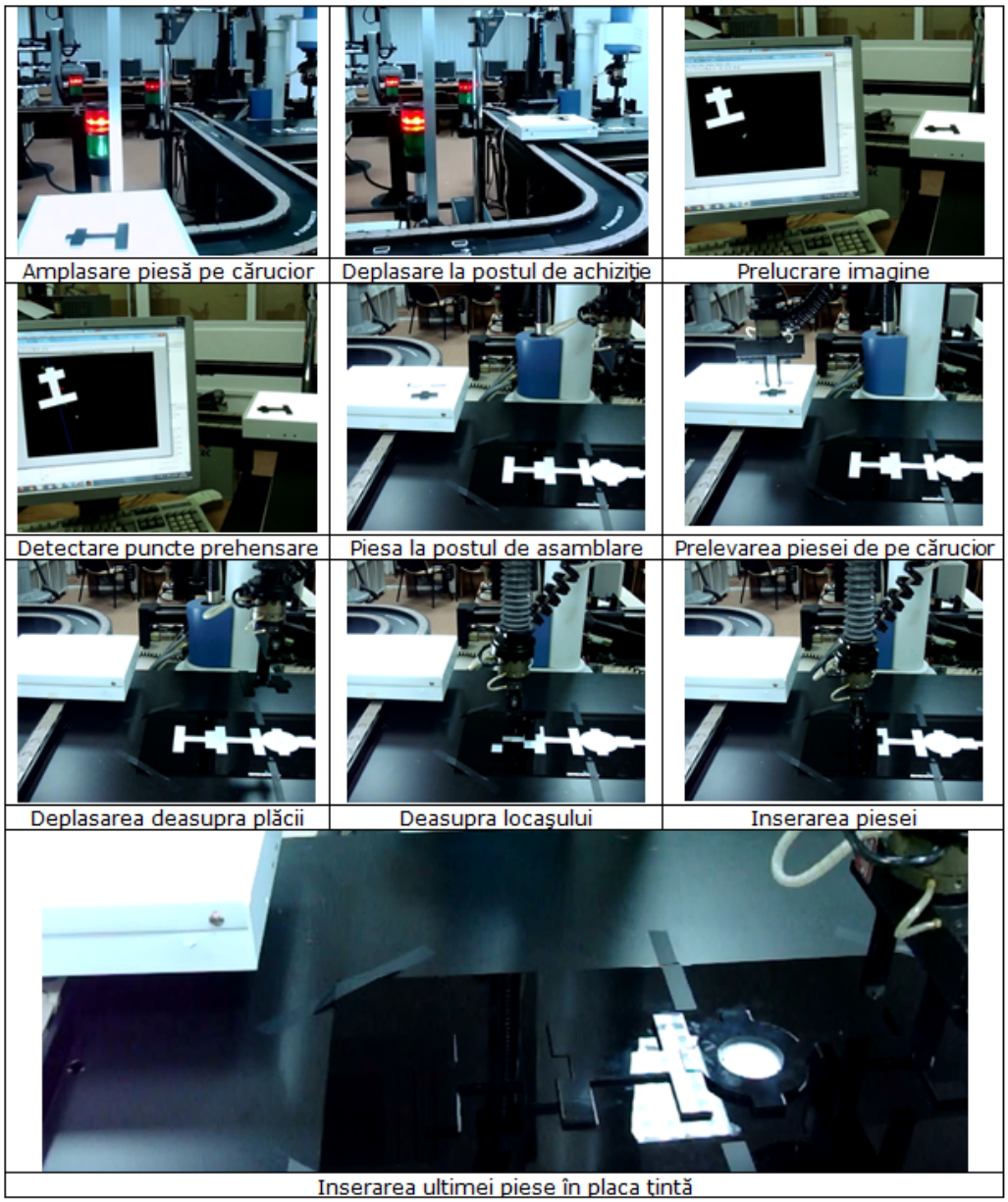


Figura 8. 12: Rezultatul rulării aplicaţiei.

9. CONCLUZII, CONTRIBUŢII ORIGINALE ŞI DIRECŢII VIITOARE DE CERCETARE

9.1. Concluzii şi contribuţii originale

În lucrarea de faţă s-a urmărit utilizarea prelucrării automate a imaginii ca parte componentă a sistemelor de vedere robotizată. Acest aspect, prezintă un interes deosebit pentru aplicaţiile industriale (subcapitolul 2.4), datorită numeroaselor avantaje obţinute, în urma implementării lor în procesul de producţie.

Studiile efectuate şi rezultatele obţinute, prezentate pe parcursul tezei, au condus la evidenţierea următoarelor contribuţii originale ale autorului:

- Realizarea unei sinteze bibliografice, a principalelor aspecte cu privire la domeniul de cercetare;
- Determinarea principalelor componente necesare în realizarea oricărei aplicaţii de vedere robotizată;
- Realizarea unei analize amănunţite a modelelor de calibrare a camerelor video, existente în literatura de specialitate;
- Implementarea în Matlab a două modele clasice de calibrare a camerelor video;
- Proiectarea, dezvoltarea şi implementarea unei aplicaţii de prelucrări de imagini, în Matlab, ce poate fi folosită la maparea unei clădiri, şi inspectarea defectelor de suprafaţă;
- Conceperea, proiectarea, şi implementarea, în Vision Builder for Automated Inspection, a unei aplicaţii de detectare şi recunoaştere a unor obiecte de tip 2D (obiecte la care cea de-a treia dimensiune este neglijabilă în raport cu celelalte două).
- Proiectarea, dezvoltarea şi implementarea unei aplicaţii de conducere a unui robot cartezian pe baza informaţiilor furnizate de un algoritm de prelucrare de imagini, utilizând mediul de lucru LabView.
- Realizarea unei aplicaţii de vedere robotizată, pentru detectarea, recunoaşterea şi manipularea stabilă a unor obiecte colorate.
- Realizarea unei analize, asupra unui metode de calibrare a camerelor video, implementată în Matlab („camera calibration toolbox", de tip „open source") şi aplicarea acesteia în calibrarea camerelor din dotare.
- Implementarea unei metode, pentru determinarea matricei de transformare omogenă, în cazul robotului SCORA ER 14.
- Dezvoltarea unui algoritm de recunoaştere a unor obiecte de tip industrial (rulmenţi radiali cu bile).
- Dezvoltarea unui algoritm original de comparare a unui obiect detectat cu un şablon;
- Proiectarea, dezvoltarea şi implementarea unei aplicaţii de vedere robotizată, pe staţia avută în dotare (în cadrul laboratorului CIM), pentru recunoaşterea unui obiect aflat într-o poziţie fixă, cunoscută.
- Dezvoltarea unui model matematic original, pentru determinarea poziţiei unui obiect în spaţiu, pe baza coordonatelor a patru puncte extrase din două imagini ale obiectului, ce sunt prelevate de două camere situate la 90^0

(unghiul format de axele optice ale celor două camere este de aproximativ 90^0).

- Proiectarea, dezvoltarea şi implementarea unei aplicații de vedere robotizată, pentru detectarea, recunoaşterea, manipularea, sortarea şi depozitarea unui obiect aflat într-o poziţie necunoscută.
- Dezvoltarea unui algoritm de detectare, decelare şi selectare a unor obiecte ce se regăsesc suprapuse într-o imagine.
- Proiectarea, dezvoltarea şi implementarea unei aplicații de vedere robotizată, pentru detectarea, recunoaşterea, manipularea, sortarea şi depozitarea mai multor obiecte suprapuse, aflat în poziții necunoscute.
- Dezvoltarea unui algoritm de recunoaştere a unor obiecte cu forme variabile.
- Implementarea unui algoritm de detectare a punctelor de prehensare optime, ale unor piese necunoscute, într-o aplicaţie robotizată, în vederea determinării orientării piesei şi realizării unei manipulări stabile, pentru efectuarea unei operaţii de asamblare.

În concluzie, se poate spune că obiectivul central al tezei (stabilit inițial), a fost îndeplinit, mai precis dezvoltarea unor aplicații de vedere robotizată, bazate pe algoritmi de prelucrare automată a imaginilor. Obiectivele conexe ale acestuia, respectiv detectarea şi recunoaşterea obiectelor de natură 2D şi 3D precum şi determinarea unei metode de stabilire a punctelor de prehensare, optime, a unor obiecte necunoscute, sau concretizat prin dezvoltarea algoritmilor prezentaţi în teză şi implementaţi în cele două aplicaţii de prelucrare automată a imagini (subcapitolul 4.2 şi 5.1) şi cele şase aplicaţii de vedere robotizată (subcapitolul 5.2, 6.1, 7.1, 7.2, 7.3, 8.1).

Contribuţiile au fost diseminate într-un număr de cinci articole ştiinţifice (vezi anexa 11):

- Două lucrări ştiinţifice publicate în volumele unor manifestări ştiinţifice (Proceedings) indexate ISI: (Pop, Grigorescu, Davidescu, 2012 a) şi (Pop, Grigorescu, Davidescu, 2012 b).
- Trei lucrări ştiinţifice publicate în reviste de specialitate indexate BDI: (Pop, Davidescu, 2010), (Pop, Davidescu şi Moldovan, 2011) şi (Pop, 2011).

9.3. Direcţii viitoare de cercetare

Pe baza cercetărilor efectuate şi a rezultatelor obţinute se pot estima câteva direcţii viitoare de cercetare:

- Proiectarea şi dezvoltarea unei aplicaţii de tip „bin-picking” care să permită detectarea, recunoaşterea şi extragerea unor obiecte suprapuse, aflate într-un container de piese.
- Conceperea, proiectarea, dezvoltarea şi implementarea unei metode automate de calibrare a camerei prin deplasări controlate ale acesteia (autocalibrare).
- Dezvoltarea unor algoritmi ce permit obţinerea unor modele cât mai realiste ale unor obiecte din lumea reală, respectiv algoritmi de recunoaştere şi reconstrucţie a unor obiecte 3D pentru aplicaţii de tip CAD.

Bibliografie

Agrawal M., Davis L. S. (2003) – „Camera calibration using spheres: A semi-definite programming approach", IEEE International Conference on Computer Vision 2003, ISBN 0-7695-1950-4, Volume 2, pp. 782-789.

Alexander M. (2010) –„An Introduction to Object Recognition", Advances in Computer Vision and Pattern Recognition, Springer, 1st Edition, XVIII, ISBN 978-1-84996-235-3.

Allen P., Timcenko A., Yoshimi B., et al. (1993) – "AutomatedTracking and Grasping of a Moving Object with a Robotic Hand-Eye System", Proceedings of the IEEE Transactions on Robotics and Automation, Vol. 9, No. 2.

Amit, Y. (2002)- „ 2D object detection and recognition: models, algorithms, and networks", MIT Press.

Annadurai S., Shanmugalakshmi R. (2007) – "Fundamentals Of Digital Image Processing", Pearson Education India, ISBN 9788177584790.

Armangue X., Salvi J., Balle J. (2000) - „A comparative review of camera calibrating methods with accuracy evaluation", Proceedings of 5th Ibero-American Symposiumon Pattern Recognition, SIAPR 2000, Lisboa, Portugal, pp. 183.194, 2000.

Austin, W. (2005) – „Inspection System Improves Diesel Engine Assembly". Assembly Magazine. Internet, BNP Media.

Automatica (2012) – „Exhibitor database AUTOMATICA 2010". Disponibil la: http://www.automatica-munich.com/exvi/en/ IBG_Automation_GmbH/DE/A2/520. Accesat în 02.05.2012.

Azernikov S. (2008) – "Sweeping solids on manifolds. In Symposium on Solid and Physical Modeling, pp. 249–255.

Bareto J. P. (2003) - „General central projection systems, modeling, calibration and visual servoing", PhD thesis, University of Coimbra.

Bay H., Ess A., Tuytelaars T., Van Gool L., (2008). „SURF: Speeded Up Robust Features", Computer Vision and Image Understanding (CVIU), Vol. 110, No. 3, pp. 346--359.

Bay H., Tuytelaars T., and Van Gool L., (2006) - „SURF: Speeded up robust features", In ECCV.

Beardsley P., Murray D., Zisserman A. (1992) – „Camera Calibration Using Multiple Images", Proceedings of the 2nd European Conference on Computer Vision, pp. 312–320.

Blake A. (1995) - "A symmetry theory of planar grasp", International Journal of Robotics Research, vol. 14, no. 5, pp. 425-444.

Bouguet J. Y. (2012) – "Camera Calibration Toolbox for Matlab". Disponibil online la: http://www.vision.caltech.edu/bouguetj/calib_doc/ . Accesat în 09.01.2012.

Bradski G., A. Kaehler (2008) - Learning OpenCV: Computer Vision with the OpenCV Library, O'Reilly Media, Incorporated.

Briechle K., Hanebeck U. D., (2001). Template matching using fast normalized cross correlation. In: Proceedings of SPIE, V. 4387, Optical Pattern Recognition XII, Orlando, FL, pp. 95-102.

Britton, D. (2011)- „Innovative Robot Uses 3D Imaging and a Novel Cutting Approach to Automatically Debone Poultry". Poultry Tech 23.

Byröd M., Kukelova Z., Josephson K., Pajdla T., Åström K. (2008) – „Fast and robust numerical solutions to minimal problems for cameras with radial distortion". In Proc. Conf. Computer Vision and Pattern Recognition, Anchorage, USA.

Cany J. (1986) –"A computational approach to edge detection", Pattern Analysis and Machine Intelligence, IEEE Transactions on PAMI-8(6), Nov. 1986,pp. 679–698.

Cao X., Shah M. (2005) - „Camera Calibration and Light Source Estimation from Images with Shadows", IEEE Conference, Computer Vision and Pattern Recognition, ISBN: 0-7695-2372-2, Volume 2, pp. 918-923.

Chang, D. Q. (2006) - "A Generative-Discriminative Hybrid Method for Multi-View Object Detection." Proceeding of IEEE International Conference of Computer Vision and Pattern Recognition, pp. 2017 - 2024

Chen I. M., Burdick J. W. (1993)- "Finding antipodal point grasps on irregularly shaped objects", IEEE Transactions on Robotics and Automation, vol. 9, no. 4, pp. 507-512.

Chen J. M., Ventura J.A., Wu C.-H. (1996) - „Segmentation of Planar Curves into Circular Arcs and Line Segments", Image and Vision Computing, 14(1), pp.71–83.

Chen X., Yang J., Waibel A. (2003) - „Calibration of a Hybrid Camera Network", IEEE International Conference on Computer Vision, ISBN 0-7695-1950-4, Volume 1, pp. 150-155.

Cherubini A., Chaumette F., Oriolo G. (2008) – "A position-based visual servoing schemefor following paths with nonholonomic mobile robots", 2008 IEEE/RSJ International Conference on Intelligent Robots and Systems Acropolis Convention Center, Nice, France, Sept, 22-26, pp. 1648-1654.

Chesi, G., K. Hashimoto (2010) – „Visual Servoing Via Advanced Numerical Methods", Springer.

Chevallet J. P., Hwee Lim J., Kew Leong M. (2005) –"Object Identification and Retrieval from Efficient Image Matching: Snap2Tell with the STOIC Dataset", Springer Verlag Berlin, AIRS 2005, LNCS 3689, pp. 97–112.

Cipolla R., Drummond T. W., Robertson D. (1999) – „Camera calibration from vanishing points in images of architectural scenes", Proceedings of British Machine Vision Conference, vol. 2, Septembre 1999, UK, pp.382-391.

CogniTens (2009) - "Ogihara Eliminates Hundreds of Inspection Hours with OptiCell by CogniTens." Disponibil la : http://www.hexagonmetrology.com/ogihara-america-howell-manual_297.htm. Accesat 04.03.2012.

Colestock, H. (2005) – „Industrial robotics: selection, design, and maintenance", McGraw-Hill.

Copot C., Burlacu A., Lazăr C. (2011) – „Image Moments based Predictive Control for Eye-in-Hand Servoing Systems", Buletinul Institutului Politehnic din Iaşi, Secţia Automatică şi Calculatoare, Fasc. 1, pp. 23-37, ISSN 1220-2169.

Corke P. I. (1996) – "Visual control of robots:High-Performance Visual Servoing", CSIRO Division of Manufacturing Technology, Australia, disponibil la : http://www.cat.csiro.au/dmt/programs/ autom/pic/book.htm. Accesat în 20.09.2011.

Corke P. (2011) – „Robotics, Vision and Control: Fundamental Algorithms in MATLAB", Springer.

Costantini G., Casali D. (2007) - "Detection of symmetry axis by a CNN-based algorithm", in*Proceedings of the 11-th WSEAS International Conference on Circuits*, AgiosNicolaos, Greece, July 22-25, pp. 46-49.

Cowan N. (2002) – „Binocular Visual Servoing with a Limited Field of View", In Mathematical Theory of Networks and Systems, Notre. Disponibil la: http://citeseerx.ist.psu.edu/showciting?doi=10.1.1.13.2088. Accesat în 01.02.2011.

Csurka G., Zeller C., Zhang Z., Faugeras O. (1987) - „Characterizing the uncertainty of the fundamental matrix. Computer Vision and Image Understanding, 68(1), Oktober 1997, pp.18–36.

Dalsa (2012) – Disponibil la http://www.teledynedalsa.com/. Accesat în 11.10.2012.

Datta A., Kim J. S., Kanade T. (2009) – „Accurate Camera Calibration using Iterative Refinement of Control Points", IEEE Conference, Computer Vision, ISBN 978-1-4244-4442-7, pp. 1201-1208.

Davidescu A. (2003) – „Analiza şi procesarea datelor în Matlab". Editura Politehnica, Timişoara.

Deutscher J., Isard M., MacCormick J. (2002) – „Automatic Camera Calibration from a Single Manhattan Image”, The 7th European Conference on Computer Vision Copenhagen, Volume 4, pp. 175-188.

Devernay F., Faugeras O. (2001) - „ Straight lines have to be straight: automatic calibration and removal of distortion from scenes of structured environments”, Machine Vision and Application, 13(1), pp.14 24.

Downhill (2012) – Disponibil la http://www.ride-downhill.de/blog/?p=708. Accesat în 01.09.2011.

Drummond T., Cipolla R. (2002) – „Real-Time Tracking of Complex Structures with on-line Camera Calibration”, Image and Vision Computing, Volume 20, Issues 5-6, pp. 427-433.

Edmund_Optics (2012) - "Electronic Imaging Resource Guide”. Disponibil la: from http://www.edmundoptics.com/learning-and-support/technical/learning-center /application-notes/imaging/electronic-imaging-resource-guide/?&pagenumElkmann

N., Lucke M., Krüge T., Kunst D., Stürze T., Hortig J. (2008) - "Kinematics, sensors and control of the fully automated façade-cleaning robot SIRIUSc for the Fraunhofer headquarters building, Munich", Industrial Robot: An International Journal, Vol. 35 Iss: 3 pp. 224 - 227

Faugeras O. (1996) –„ Three-Dimentional Computer Vision. A Geometric Viewpoint”, MIT Press, Cambridge, Massachusetts.

Faugeras O. Toscani G. (1987) – „Camera Calibration for 3D Computer Vision”, Proceedings ofInternational Workshop on Industrial Applications of Machine Vision and Intelligence, Tokyo, Japan, 1987, pp. 240-247.

Faugeras O., Toscani G. (1986) – „The calibration problem for stereo”, Proceedings CVPR’86, Miami Beach, Florida, June 1986, pp.15-20.

Fiala M., Shu C. (2010) – „Fully Automatic Camera Calibration Using Self-Identifying Calibration Targets”, National Research Council of Canada, disponibil la: http://nparc.cisti-icist.nrc-cnrc.gc.ca/npsi/ctrl?action=shwart&index=an&req=8913 774& lang = en. Accesat în 13.12.2010.

Forsyth, D. A., Ponce J. (2011) - „Computer Vision: A Modern Approach”, Prentice Hall PTR.

Fuageras O., Luong T. Q., Maybank S. (1992) - „Camera self-calibration:theory and experiments”, Proceedings of the 2nd European Conference on Computer Vision, Santa Margherita, Italy, May, 1992, pp.858-863.

Fuentes-Pacheco J., Ruiz-Ascencio J., Rendón-Mancha J. M. (2009) – “Binocular visual tracking and grasping of a moving object with a 3D trajectory predictor”, Journal of Applied Research and Technology, Vol.7 No. 3 December 2009, pp. 259-274.

Fung G., Yung N., Pang G. (2003) - „Camera calibration from road lane markings“, Optical Engineering, Volume 42, Issue 10, pp. 2967-2977.

Gans N. R. (2003) – „An experimental study of hybrid switched system approaches to visual servoing”, Robotics and Automation, 2003. Proceedings. ICRA '03. IEEE International Conference on 14-19 Sept. 2003, vol. 3, pp. 3061 – 3068.

Glaser, A. (2008) – „Industrial robotics: how to implement the right system for your plant”, Industrial Press.

Goldstein H., Poole C., Safko J. (2002) – “Classical Mechanics”, 3rd ed., San Francisco, USA: Addison Wesley.

Gonzalez, R. C. (2002) - „Digital Image Processing”, New Jersey, Prentice Hall.

Grigorescu S. M. (2012) – „Lucrare laborator CIM”. Departamentul de Mecatronică, Universitatea Politehnica Timişoara.

Gruescu C. (2012) – “Curs senzori vizuali”. Departamentul de Mecatronică, Universitatea Politehnica Timişoara.

GTRI (2012) - "A Cut Above: Innovative Robot Uses 3-D Imaging and Sensor-based Cutting Technology to Debone Poultry ". Disponibil la: from http://www.gtri.gatech.edu/casestudy/robot-3d-imaging-sensor-based-debone-poultry. Accesat 05.04. 2012.

Gümüş B., Balaban M. O., Ünlüsayın M. (2011) – “Machine Vision Applications to Aquatic Foods: A Review”, Turkish Journal of Fisheries and Aquatic Sciences 11, pp.171-181.

Guoshen Y., and Morel J.-M. (2009). „ASIFT: ASIFT, A new framework for fully affine invariant image comparison”, SIAM Journal on Imaging Sciences, 2(2), pp. 438-469.

Hall E., Tio J., McPherson C., Sadjadi, F. (1982) – „Measuring curved surfaces for robot vision”, Computer Journa, vol. December, 1982, pp.42-54.

Hanning T., Schoene R. (2007) - „ Additional constraints for Zhang’s closed form solution of the camera calibration problem”, Technical Report MIP-0709, Fakultät für Informatik und Mathematik, Universität Passau, 2007.

Hartenberg R., Denavit J. (1964) – „Kinematic Synthesis of Linkages”, McGraw Hill, New York, Lybrary of Congress Catalog Card Number 64-2351.

Hartley R. (1992) – „Estimation of relative camera position for un-calibrated cameras”, Proceedings of the 2nd European Conference on Computer Vision, Santa Margherita, Italy, May 1992, pp. 579-587.

Hartley R. (1994) – „Self-Calibration from Multiple Views with a Rotating Camera”, Proceedings of 3rd European Conference on Computer Vision, Stockholm, vol. 1, pp. 471-478.

Hartley R. (1997) – „ Self-Calibration of Stationary Cameras”, International Journal of Computer Vision, vol. 22, no.1, February 1997, pp. 5-23.

Hartley R. I., Zisserman A. (2004) - Multiple View Geometry in Computer Vision. Cambridge University Press, Second Edition.

Hegde, G. S. (2007) – „A Textbook on Industrial Robotics”, Laxmi Publications.

Heikkilä J. (1997) –„ Accurate camera calibration and feature based 3D reconstruction from monocular image sequences”. PhD thesis, University Oulu, Finland.

Heikkilä J. (2000) -„ Geometric camera calibration using circular control points”, IEEE Transactions on Pattern Analysis and Machine Intelligence, 22(10) pp. 1066–1077

Henry A. R., Shumeet B., Takeo K., (1995) - „Human face detection in visual scenes”, In Advances in Neural Info. Proc. Systems, volume 8.

Heyden A. , Astrom K. (1997) – „Euclidean reconstruction from image sequences with varyingand unknown focal lenght and principal point”, Proceedings of IEEE Conference on Computer Vision and Pattern Recognition, 1997, pp. 477-492.

Iordăchiţă, I. (1997) – „Roboţi industriali”, Craiova, Reprografia Universităţii din Craiova.

Jain, A. K. (1999) – „Fundamentals of Digital Image Processing”. Englewood Cliffs NJ, Prentice Hall.

Joochim C., Roth H. (2010) – „Mobile Robot Exploration Based On Three Dimension Cameras Acquisition”, presented at the 2nd IFAC Symposium on Telematics Applications - TA2010, Timişoara, România.

Keller Y., Shkolnisky Y. (2006) - “A signal processing approach to symmetry detection”, IEEE Transactions on Image Processing, vol. 15, no. 8, pp. 2198-2207.

Kim J.K. (2006) – “Image-based Visual Servoing using Sliding mode control”, SICE-ICASE, 2006. International Joint Conference, ISBN 89-950038-4-7.

Kiryati N., Gofman Y. (1998) - “Detecting symmetry in grey level images: The global optimization approach”, *International Journal of Computer Vision*, vol. 29, no. 1, pp. 29-45.

Koulkarni P., Dutta D., Saigal R. (1995) - “An investigation of techniques for asymmetry rectification”, Trans. ASME, Journal of Mechanical Design, 117, pp. 620-626.

Kovacs F. V., Varga. Şt., Pau V.C. (2000) – „Introducere în robotică”, Bucureşti, Printech.

Kragic D., Christensen H. I. (2003) –“Robust Visual Servoing”, The International Journal of Robotics Research,Vol. 22, No. 10, October 2003, pp. 1-17.

Kruachottikul, P. (2012) -"Building a Robotic Vision System for Cylinder Head Inspection". Disponibil la: http://sine.ni.com/cs/app/doc/p/id/cs-13945. Accesat în 06.05.2012.

Lenz R. K., Tsai R. Y. (1988) – „ Techniques for Calibration ofthe Scale Factor and Image Center for High Accuracy 3D Machine Vision Metrology", IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 10, no. 5, September1988, pp. 713-720.

Li W. H., Zhang A. M., Kleeman L. (2005) - "Fast global reflectional symmetry detection for robotic grasping and visual tracking", in Proceedings of Australian Conference on Robotics and Automation, ACRA05, M. M. Matthews, Ed., December.

Lowe, D. G. (2004) - "Distinctive Image Features from Scale-Invariant Keypoints" International Journal of Computer Vision 60(2), pp. 91-110.

Luong T. Q., Faugeras O. (1996) – „The Fundamental Matrix: Theory, Algorithms and Stability Analysis", International Journal of Computer Vision, vol. 17, no. 1, pp. 43-76.

Luong T. Q., Faugeras O. (1997) – „Self-calibration of a moving camera from point correspondence and fundamental matrix", International Journal of Computer Vision 22(3), pp. 261-289.

Lv F., Zhao T., Nevatia R. (2006) – „Camera Calibration from Video of a Walking Human", IEEE Pattern Analysis and Machine Intelligence, ISSN: 0162-8828, Volume 28, Issue 9, pp. 1513-1518.

Marchand E., Chaumette F. (2005) – "Feature tracking for visual servoing purposes", Robotics and Autonomous Systems 52, 1, pp. 53-70.

Marola G. (1989) - "On the detection of the axes of symmetry of symmetric and almost symmetric planar images", IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 11, no. 1, pp. 104-108.

Marita T. (2012) – „Sisteme de viziune in robotica - curs" Disponibil la: http://users.utcluj.ro/~tmarita/SVR/C1.2.pdf. Accesat 05.08.2012.

Martin R. R., Dutta D. (1996) - "Tools for asymmetry rectification in shape design", Journal of Systems Engineering, vol. 6, pp. 98-112.

Martins H. A,. Birk J. R , Kelley R. B. (1981) – „Camera models based on data from two calibration planes", Computer Graphics and Image Processing, 17:173 , pp. 180.

Mathia, K. (2010) – „Robotics for Electronics Manufacturing: Principles and Applications in Cleanroom Automation", Cambridge University Press.

MATROX (2012) - „Vision help delta robots sort biscuits", disponibil la: http://www.ukiva.org/vision-system-application-articles/matrox.html. Accesat 05.04 .2012.

Maybank S.J, Faugeras O.D (1992) - „A theory of self-calibration of a moving camera", International Journal of Computer Vision, vol 8, pp 123-152.

McCartin B. J. (2008) - "On the relationship between concentration and inertia hyperellipsoids", Applied Mathematical Sciences, vol. 2, no. 10, pp. 489-495.

Ménegaux D., Faudot D., Kheddouci H. (2004) – "Skeletizing 3D-Objects by Projections", Computational Science and Its Applications – ICCSA 2004, Lecture Notes in Computer Science, Volume 3045, 2004, pp 267-276.

Mirtich B., Canny J. (1994) - "Easily computable optimum grasps in 2-D and 3-D", IEEE International Conference on Robotics and Automation, pp. 739-747.

Montana D. J. (1992) - "Contact stability for two-fingered grasps", IEEE Transactions on Robotics and Automation, vol. 8, no. 4 August 1992, pp. 421-430.

Murase H., Nayar S. - (1995) - "Visual Learning and Recognition of 3-D Objects from Appearance", International Journal Computer Vision, 14, pp. 5–24.

Nguyen V.-D. - "Constructing force-closure grasps", International Journal of Robotic Research, vol. 7, no. 3, June 1988, pp. 3-16.

Niblack C.W., Barber R.J., Equitz W.R., Flickner M.D., Glasman D., Petkovic D. and Yanker P.C., (1993). "The QBIC Project: Querying Image by Content Using Color, Texture and Shape", In Electronic Imaging: Storage and Retrieval for Image and Video Databases, Proceedings SPIE, 1908, pp. 173–187.

Nixon, M. S., A. S. Aguado (2008) – „Feature extraction and image processing", Academic.

Nomura, H., Naito T. (2000) – "Integrated visual servoingsystem to grasp industrial parts moving on conveyer by controlling 6DOF arm", Proceedings of the IEEE International Conference on Systems, Man, and Cybernetics, pp. 1768-1775.

O'Mara D., Owens R. (1996) - "Measuring bilateral symmetry in digital images",IEEE – TENCON – Digital Signal Processing Applications, 1996, pp. 151-156.

Obinata G., Dutta A. (2007) – „Vision Systems: Applications", I-Tech Education and Publishing, ISBN 978-3-902613-01-1.

Oi J., Rao K. (1991) - New insights into correlation-based template matching. In: Proceedings of SPIE, V. 1468, Applications of Artificial Intelligence IX, Orlando, FL, pp. 740-751.

Otsu N. (1979) – "A Threshold Selection Method from Gray-Level Histograms". IEEE Transactions on Systems, Man, and Cybernetics. 1979, 9(1), pp. 62-66.

Papageorgiou C. , Poggio T. (2000) - „A trainable system for object detection", Intl. J. Computer Vision, 38(1), pp. 15–33.

Papoulis A. (1991)- "Probability, Random Variables, and Stochastic Processes", USA, New York: McGraw Hill, 3rd ed.

Perceptron (2012) - "Applications". Disponibil la: http://www.perceptron. com/index.php/en/-industrial.html. Accesat 02.03. 2012.

Petrou M. , Petrou C. (2010) – "Image Processing: The Fundamentals", John Wiley & Sons, ISBN 9780470745861.

Pollak I., Willsky A., Krim H. (2000) – „Image Segmentation and Edge Enhancement with Stabilized Inverse Diffusion Equations", IEEE TRANSACTIONS ON IMAGE PROCESSING, VOL. 9, NO. 2, FEBRUARY 2000, pp. 256-266.

Pomares J., Corrales J. A., García G. J. and Torres F. (2011) –"Direct Visual Servoing to Track Trajectoriesin Human-Robot Cooperation", Int J Adv Robotic Sy, 2011, Vol. 8, No. 4, pp. 129-138.

Pop C. (2011c) – "Improvement of Automated Robot Assembly Process using Image Processing Methods", Workshop_ul nr.2 Interdisciplinaritatea şi managementul cercetării, Universitatea „Politehnica" din Timişoara, Secţiunea Mecanică, Timişoara, pp. 45-46.

Pop C. (2010a) – „Stadiul actual al medodelor de prelucrare a imaginilor în timp real" (Referat Activităţi Complementare). Departamentul de Mecatronică, Universitatea Politehnica din Timişoara

Pop C. (2011a) – Raport stagiu. Departamentul de Mecatronică, Universitatea Politehnica din Timişoara.

Pop C., Davidescu A. (2010b) – "Buildings Inspections by Image Processing Approach", Proceedings of the Fifth International Conference on Optimization of the Robots and Manipulators, OPTIROB 2010, Călimăneşti, România, ISBN 978-981-08-5840-7, May 2010, pp. 205-209.

Pop C., Davidescu A., Moldovan F. (2011b) – "Object recognition using a smart camera", Romanian Review Precision Mechanics, Optics And Mechatronics, ISSN 1584-5982, Editura Cefin, Bucuresti, Romania, nr. 39, 2011, pp. 189-193.

Pop C., Grigorescu S. M., Davidescu A. (2012a) – "Colored object detection algorithm for visual-servoing application", Proc. 13th Int. Conf. on Optimization of Electrical and Electronic Equipment OPTIM 2012, Brasov, Romania, ISBN: 978-1-4673-1653-8/12, May 2012, IEEE , pp. 1539-1544.

Pop C., Grigorescu S. M., Davidescu A. (2012b) – "Robot Vision Application for Bearings Identification and Sorting", Mechanisms, Mechanical Transmissions and Robotics, ISBN-13:978-3-03785-395-5, Applied Mechanics and Materials, vol. 162, ISSN 1660-9336 Trans Tech Publications, pp.523-531.

Pratt W. K. (2001) - "Digital Image Processing", PIKS Inside, 3rd ed., USA, New York:John Willey & Sons.

Press W. H., Teukolski S. A., Vetterling W. T., Flannery B. P. (1992) – „Numerical Recipes in C: The Art of Scientific Computing", Cambridge University Press, New York.

Ramalingam S., Sturm P., Lodha S. (2005) – „Towards Complete Generic Camera Calibration", IEEE Computer Vision and Pattern Recognition, ISBN 0-7695-2372-2, Volume 1, pp. 1093-1098.

Ramnath K. (2004) - "A framework for robotic vision-based grasping task". Carnegie Mellon University, Pittsburgh, USA, The Robotics Institute Project Report.

Rădulescu C. (2008) – "Curs de robotică avansată",Universitatea Politehnica Timişoara.

Rekleitis I., Dudek G. (2005) – „Automated Calibration of a Camera Sensor Network", IEEE/RSJ International Conference on Intelligent Robots and Systems, ISBN 0-7803-8912-3, pp. 3384-3389.

Remazeilles A., Chaumette F., Gros P. - "Robot motion control from a visual memory", IEEE International Conference on Robotics and Automation, vol. 4, pp. 4695–4700.

Richard, G. Z. J. (2005) – „Guided by Vision. Assembly Magazine", Internet, BNP, Media: 5.

ROBOT (2012) – „iR Vision-„2D Single"". Disponibil la: http://www.robotsdotcom.com/VisionR-30iA.pdf. Accesat în 10.02.2012.

Robotics, I. F.O (2005) - „World Robotics: Statistics, Market Analysis, Case Studies And Profitability of Robot Investment", 2005, UN.

Romeo, I. (1996) – „Roboţi Industriali", Bucureşti.

Rosin P.L., (1997) - „Techniques for assessing polygonal approximations of curves", IEEE Transactions Pattern Analysis and Machine Intelligence, vol. 19, no. 6, pp. 659-666.

Russ, J.C. (2007**)** – „The Image Processing Handbook". Fifth Edition. s.l. : CRC Press, ISBN 0-8493-7254-2.

Sanz P. J. (2005) - "Tutorial: Towards autonomous manipulation capabilities for service robots",IEEE International Conference on Mechatronics and Automation, ICMA 2005, Niagara Falls, Canada.

Savii G. G. (2004) – „Camera Calibration Using Compound Genetic-Simplex Algorithm", Journal of Optoelectronics and Advanced Materials Vol. 6, No. 4, December 2004, pp. 1255 – 1261.

Schneiderman H., Takeo K., (2000) - „A statistical model for 3D object detection applied to faces and cars", In CVPR.

Secuv (2012) –„ CCD Format Size”, disponibil la: http://www.secuv.com/CCD-Format-Size. Accesat în 10.10.2011.

Seeman T. (2002) –“Digital Image Processing Using Local Segmentation”, PhD, disponibil la: http://www.csse.monash.edu.au/~torsten/pubs/Seemann-thesis.pdf. Accesat în 12.10.2011.

Shah S., Aggarwal J. K. (1996) – „Intrinsic Parameter Calibration Procedure for a (High-Distortion) Fish-Eye Lens Camera with Distortion Model and Accuracy Estimation”, Pattern Recognition, Volume 11, pp. 1775-1788.

Sheehy D.J., Armstrong C.G., Robinson D.J. (1996) – „Shape Description By Medial Surface Construction”, IEEE Trans. on Visualization and Computer Graphics 2(1), pp. 62–72.

Shi Y., Real F. D. (2010) – “Smart Cameras: Fundamentals and Classification”, Springer, ISBN 978-1-4419-0952-7, pp. 19-34.

Shih S.W., Hung Y.P., Lin W.S. (1993) – „Accurate Linear Technique for Camera Calibration Considering Lens Distortion by Solving an Eigenvalue Problem”, Optical Engineering, Volume 32, Issue 1, pp. 138-149.

Shimshoni I., Basri R., Rivlin E. (1999) - A Geometric Interpretation ofWeak-Perspective Motion, IEEE Transactions on pattern analysis and machine intelligence, vol. 21, no. 3.

SIRTKAYA, S. (2004) – „Moving Object Detection in 2D and 3D Scenes”, M.S., Department of Electrical and Electronics Engineering, Middle East Technical University MASTER: 88.

SOLIMAC (2012) – „Robot Vision System”, Disponibil la: http://www.Solimac automation.com/. Accesat în 01.05.2012.

Solomon C., Breckon T. (2011) - “Fundamentals of Digital Image Processing. A Practical Approach with Examples in Matlab”, UK, Oxford: Wiley-Blackwell.

Sonka M., Hlavac V., Boyle R. (2008) – “Image Processing, Analysis and Machine Vision”, Toronto, Canada: Thomson, 2008.

Spring K. R. (2002) – „Electronic Imaging in Neuroscience”, Current Protocols in Neuroscience. 2.4.1–2.4.9. Disponibil la http://onlinelibrary.wiley.com /doi/ 10.1002/ 0471142301.ns0204s18/full. Accesat 20.11.2012.

Sturm P. (1997) – „Vision 3D Non Calibree:Contributions a la reconstruction projective et etude des mouvements critiques pour l’auto-calibrage”, PhD l’Institut National Polytechnique de Grenoble, France, 1997.

Sturm P. (2002) – „ Critical Motion Sequences for Self-Calibration of Cameras and Stereo Systems with Variable Focal Lenght”, Image and Vision Computing, no. 5-6, vol. 20, March 2002, pp. 415-426.

Sturm P. F , Maybank S. J (2010) – „ On plane-based camera calibration: a general algorithm, singularities, applications", In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), volume 1, pp. 432–437.

Sturm P., et al. (2011) - Camera Models and Fundamental Concepts Used in Geometric Computer Vision, Foundations and Trends in Computer Graphics and Vision archive, Volume 6 Issue 1–2, Now Publishers Inc. Hanover, MA, USA.

Sutton M., Orteu J.J., Schreier H. (2009) – „Image Correlation for Shape, Motion and Deformation Measurements", Springer, ISBN 978-0-387-78746-6.

Symon K. R. (1971) – "Mechanics", 3rd ed., Reading, USA: Addison-Wesley.

Taylor G., Kleeman L. (2006) – „Visual Perception and Robotic Manipulation", Springer Tracts in Advanced Robotics, Vol. 26, ISBN 978-3-540-33455-2.

Telea, D. C., A.N. (2002) - „Roboţi", Cluj Napoca, Editura Dacia.

Tell D. (2002) – „Wide baseline matching with applications to visual servoing", PhD thesis, Computational Vision and Active Perception Laboratory (CVAP), Royal Institute of Technology, June,Stockholm, Sweden.

ThomasNet (2012) – "PC-based Machine Vision Versus Smart Camera Systems." Disponibil la: http://www.thomasnet.com/articles/automation-electronics/smart-camera-versus-pc-based-machine. Accesat 03.10.2012.

Torralba A., Fergus R., Freeman W. T., (2008) - „80 million tiny images: a large data set for nonparametric object and scene recognition", IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 30, no. 11, pp. 1958–70.

Tsai D.-M., Tsai Y. H., (2002) – „Rotation-invariant pattern matching with color ring-projection". Pattern Recognition 35, 131-141.

Tsai R. Y. (1986) – „An Efficient and Accurate Camera Calibration Technique for 3D Machine Vision", Proceedings CVPR'86, Miami Beach, Florida, June 1986, pp. 364-374.

Tsai R. Y. (1987) – „ A Versatile Camera Calibration Technique for High-Accuracy 3D Machine Vision Metrology Using Off-the-Shelf Cameras and Lenses", IEEE Journal of Robotics and Automation, vol. RA-3, No.4, August 1987, pp. 323-344.

Tzimiropoulos G., Agryriou V., Stathaki T. (2008) - "Symmetry detection using frequency domain motion estimation techniques", IEEE International Conference on Acoustics, Speech and Signal Processing, ICSAP 2008, pp. 861-864.

Ude A. (2010) – „Robot Vision", InTech, ISBN 978-953-307-077-3.

Viola P., Jones M., (2004) - „Robust real-time object detection", Intl. J. Computer Vision, 57(2), pp.137–154.

VISION, E. S. M. (2012) - "Robot guided valve inspection". Disponibil la : http:// www.esmvision.eu/cognex/application/application-in-your-industry/automotive/ automotive-page-9/robot-guided-valve-inspection/. Accesat 03.03. 2012.

VISION, E. S. M. (2012) - "High speed bin-picking". Disponibil la :http://www. esmvision.eu/cognex/. Accesat în 05.01.2011.

Vlaicu, A. (1997) –„Prelucrarea digitală a imaginilor”, Cluj Napoca, Editura Albastră

Vona M., Quigley K., Rus D. (2010) – “Eye-In-Hand Visual Servoingwith a 4-Joint Robot Arm. An Introductory Robotics Workshop at MIT CSAIL”, disponibil la: http:// groups.csail.mit.edu/ drl/wiki/ images/f/f2/VisualServoing.pdf. Accesat în 02.10.2012.

Wang L., Shi J., Song G., Shen I., (2007) - „Object Detection Combining Recognition and Segmentation”, ACCV 1, Vol. 4843 Springer, pp. 189-199.

Wang L.L., Tsai W.H. (1991) – „Camera Calibration by Vanishing Lines for 3-D Computer Vision”, IEEE Transactions, Pattern Analysis and Machine Intelligence, ISSN 0162-8828, Volume 13, Issue 4, Pag. 370-376.

Wei G.-Q,. Ma S. D-, Implicit and explicit camera calibration:Theory and experiments”, IEEE Transactions on Pattern Analysis and Machine Intelligence, 16(5), pp. 469 – 480.

Weng J., Cohen P., Herniou M. (1992) – „ Camera calibration with distorsion models and accuracy evaluation”, IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 14, 1992, pp. 965-980.

Wernholt, E. (2007) –„Multivariable Frequency-Domain Identification of Industrial Robots”, Department of Electrical Engineering. Linköping, Sweden, Linköping University. Dissertation: 224.

Wong K. Y., Mendonca P., Cipolla R. (2003) – „Camera Calibration from Surfaces of Revolution”, IEEE Transactions, Pattern Analysis and Machine Intelligence, ISSN 0162-8828, Volume 25, Issue 2, pp. 147 – 161.

Yang J., Zhang M., Wang Y., Shang Y. (2007) – “A Monocular Visual Servoing Control System for Mobile Robot”, Automation and Logistics, 2007 IEEE International Conference on Topic(s) Computing & Processing (Hardware/Software); Robotics & Control Systems; Signal Processing & Analysis; Transportation, pp. 574 – 579.

Yu Shi, Real F. B. D. (2010) - „Smart Cameras: Fundamentals and Classification”, Springer, ISBN 978-1-4419-0952-7, pp. 19-34.

Zamperoni, P. (1995) – „Image Enhancement”, Advances in Imaging and Electron Physics 92, pp.1-77.

Zhang Z. (2004) – „Camera Calibration With One-Dimensional Objects”, IEEE Transactions, Pattern Analysis and Machine Intelligence, ISSN 0162-8828, Volume 26, Issue 7, pp. 892-899.

Zhang Z. (2008) – „A Flexible New Technique for Camera Calibration”, IEEE Transactions, Pattern Analysis and Machine Intelligence, ISSN 0162-8828, Volume 22, Issue 11, pp. 1330 – 1334.

Zielke T., Brauckmann M., Von-Seelen W. (1993) - „Intensity and edge based symmetry detection with application to car following”, *CVGIP: Image Understanding*, vol. 58, pp. 177-190.

Zisopol, D. G. (2006) - „ Roboţi industriali”, Ploieşti, Editura Universităţii Petrol-Gaze din Ploieşti.

Anexe

Anexa 1: Evoluţia roboţilor industriali

În anul 1954 George Devol a proiectat primul robot cu acţionare hidraulică, iar ulterior în 1956 împreună cu Joseph Engelberger înfiinţează „Unimation INC." prima firmă de fabricare a roboţilor. Primul produs al acestei firme este un prototip de braţ robotic instalat în cadrul fabrici General Motors în 1959, iar în 1961 este comercializat în urma patentării şi este utilizat în cadrul aceleiaşi firme ca robot industrial într-o linie de asamblare pentru preluarea de semifabricate de la o maşina de turnat sub presiune (Kovacs F. V. 2000), (Mathia 2010).

În 1960 proiectanţii Harry Johnson şi Veljko Milenkovic realizează pentru corporaţia americană AMF primul robot în coordonate cilindrice cu acţionare pneumatică numit Versatran (Kovacs F. V. 2000), (Mathia 2010).

În 1968 la institutul de cercetare de la Stanford (SRI - Standford Researc Institute) este construit un robot mobil dotat cu vedere artificială numit „Shakey" ce este comandat de un calculator de dimensiunea unei camere. Anul următor profesorul Victor Scheinman realizează un braţ robotic numit „Stanford Arm" acţionat electric şi comandat de un calculator (Telea 2002).

În 1973 firma KUKA din Germania dezvolta primul robot cu şase axe acţionate electromecanic numit „Famulus", iar firma japoneză Hitachi realizează primul robot cu vedere artificială utilizat pentru obiecte aflate în mişcare (înşuruba şi deşuruba şuruburi în piese) (Zisopol 2006).

Anul următor la firma Cincinnati Milacron Corporation este dezvoltat de către Richard Hohn robotul T3 intitulat „The Tomorrow Too". Acesta este cu acţionare hidraulică şi controlat cu ajutorul unui minicomputer (Robotics 2005).

În 1974 sunt produşi primii roboţi industriali cu acţionare electrică integrală de firmă suedeză ASEA. Controlerul S1 al acestor roboţi a fost primul care a utilizat un microprocesor Intel pe 8 biţi. Primul model al acestei familii de roboţi „IRB6" a fost utilizat de firma Magnussons pentru ceruirea şi lustruirea unor ţevi din oţel inoxidabil (Zisopol 2006).

În 1978 firma Vicarm a lui Victor Scheinman preluată de Unimation dezvoltă robotul PUMA (Programmable Universal Machine for Assembly) pentru General Motors care îl utilizează la o linie de producţie pentru manipularea unor piese de dimensiuni reduse (Mathia 2010).

Anul următor Hiroshi Makino, de la Universitatea Yamanashi din Japonia, dezvoltă robotul SCARA (Selective Compliance Assembly Robot Arm –braţ robotic pentru asamblare cu complianţă selectivă) (Kovacs F. V. 2000).

În România în 1980 apare primul robot industrial „Rip 6.3" de tip ASEA construit la întreprinderea Automatica Bucureşti sub coordonarea I.C.P.T.C.M. Bucureşti în cadrul unui program naţional (Iordăchiţă 1997).

În 1981 este realizat primul braţ articulat robotic de tip portal de către firma PaR Systems din Statele Unite ale Americi (SUA) (Zisopol 2006).

În anul 1982, la Timişoara, în cadrul institutului Politehnic „Traian Vuia" apare primul robot de concepţie integrală românească „REMT-1" realizat de către colectivul de robotică condus de regretatul profesor Francisc Kovacs. Acest robot a fost folosit la Electromotor Timişoara în cadrul unei celule de prelucrare prin aşchiere a unei familii de arbori (Richard 2005), (Kruachottikul 2012).

În 1984 apare „AdeptOne" primul robot de tip SCARA cu acţionare directă utilizând doar motoare electrice (lipsesc transmisiile mecanice din cadrul sistemului de acţionare) realizat de firma Adept din SUA.

În anul 1985 firma KUKA introduce un nou tip de robot industrial a cărui formă este de tip „Z" (Zisopol 2006).

În 1987 Institutul Politehnic „Traian Vuia" Timişoara şi IMMUM Baia Mare realizează un manipulator sincron – MS 500 expus la TIB´87 şi distins cu Medalia de aur iar anul următor realizează un robot portal „ROPOS 50" Expus la TIB´88 şi distins cu tot Medalia de aur (Iordăchiţă 1997).

În 1998 firma suedeză ABB realizează, pe baza proiectului lui Reymond Clavel de la Federal Institute of Technology of Lausanne (EPFL), cel mai rapid robot utilizat în operaţiile industriale de tip „Pick and Place". Robotul intitulat „FlexPicker" este capabil să apuce 120 de obiecte într-un minut respectiv să manipuleze (apucare/eliberare) obiecte cu o viteză de 10m/s utilizând un sistem de vedere artificială (Zisopol 2006).

În 2006 este dezvoltat de către firma germană KUKA în colaborare cu Institutul de Robotică şi Mecatronică din Germania primul robot de tip "Light Weight Robot". Robotul este unul portabil datorită faptului că are o greutate de 16 kg, structura sa externă fiind realizată din aluminiu, şi este capabil să ridice obiecte de până la 7 kg (Zisopol 2006).

În 2010 firma japoneză Fanuc lansează primul robot dotat cu programul de învăţare şi control a vibraţiilor (Learning Vibration Control Software - LVC) care conferă robotului o precizie în poziţionare ridicată datorită nivelului de vibraţii scăzut la viteze şi acceleraţii mari.

Anul următor, în 2011, firma suedeză ABB lansează robotul industrial Frida prevăzut cu doua braţe flexibile, sistem de vedere artificial şi un controler inteligent de tip IRC5 destinat să fie utilizat pentru manipularea pieselor mici în sistemele de fabricaţie flexibile.

Anexa 2: Codul sursă, în Matlab, pentru metoda Tsai de calibrare a camerelor video.

```
% #################Cristian Pop - student Drd #################
%                 CALIBRAREA UNEI CAMERE VIDEO
%                   pe baza metodei lui Tsai
%
% Date de intrare:
% * Coordonatele 3D ale punctelor raportate la un sistem de referinta
% global = fisier text "CoordonateA8Pct3D.txt".
% * Coordonatele 2D ale punctelor raportate la sistemul de referinta al
% imaginii = fisier text "CoordonateA8Pct2D.txt".
% * Se presupune centrul imaginii cunoscut.
% Date de iesire:
% * Parametri extrinseci ai camerei video (matricea de rotatie si de
% translatie).
% * Parametri intrinseci ai camerei video (distanta focala, raportul de
% aspect).
#########################################################
%%
function [MatRotatie, MatTranslatie,DistantaFocala] = CalibrareMD7(~,
~)
% Icarcarlea fisierul text cu coordonatele 3D ale punctelor
[MatPct3D]=textread('CoordonateA8Pct3D.txt');
% Determina marimea matrici coordonatelor 3D
[NrRand3D, NrCol3D]=size(MatPct3D);
% Verifica daca matricea este de ordinul nx3
if(NrCol3D~=3)
    error('Matricea coordonatelor 3D nu este formata din trei
coloane.');
end
Xw = MatPct3D(:,1); % Extrage doar valorile X din matrice
Yw = MatPct3D(:,2); % Extrage doar valorile Y din matrice
Zw = MatPct3D(:,3); % Extrage doar valorile Z din matrice
%% Pct2D
% Citeste fisierul text cu coordonatele 2D
[MatPct2D]=textread('CoordonateA8Pct2D.txt');
 % Determina marimea matrici 2D
[NrRand2D, NrCol2D]=size(MatPct2D);
% Verifica daca matricea este de ordinul nx2
if(NrCol2D~=2)
    error('Matricea coordonatelor 2D nu este formata din doua
coloane.');
end
xim=MatPct2D(:,1); % Extrage doar valorile x din matrice
yim=MatPct2D(:,2); % Extrage doar valorile y din matrice
%% VerificareRanduri
% Se verfica daca numarul randurilor matrici 2D este egal cu numarul
% randurilor matrici 3D
if(NrRand3D~=NrRand2D)
    error('Verifica ca numarul punctelor 2D si 3D sa fie acelasi.');
end
%% Matricea A
```

```
% Construieste matricea A corespunzatoare sistemului omogen de N
ecuatii
% liniare. Daca N>=7, iar punctele N nu sunt coplanare, rangul lui A
este 7
% iar sistemul Av=0 are o solutie netriviala(nebanala).
A=[xim.*Xw xim.*Yw xim.*Zw xim -yim.*Xw -yim.*Yw -yim.*Zw -yim];
%% SVD din A
% Descompuneraea Matrici A utilizand metoda "Descompunerea Valorilor
% Singulare".
[S M D]=svd(A) % S=stanga, M=mijloc, D=dreapta
%% Vectorul v
% Definire vectorului celor opt necunoscute independente.
v=D(:,end);
v1=v(1,1);
v2=v(2,1);
v3=v(3,1);
v5=v(5,1);
v6=v(6,1);
v7=v(7,1);
%% Gama si alpha
% Determinarea factorului de scalare si a raportului de aspect.
gamma=sqrt(v1^2+v2^2+v3^2);
alpha=sqrt(v5^2+v6^2+v7^2)/gamma;
%% Determinare partiala R si T
% Pe baza celor cunoscute pana in momentul de fata putem determina
primele
% doua randuri ale matrici de rotatie R si primii doi termeni ai
vectorului
% de translatie T
% Construieste o matrice de rotatie de tip 3x3  care sa fie populata cu
% elemente cu valoarea zero
R=zeros(3,3);
R(1,:)=v(5:7,1)/(alpha*gamma);
R(2,:)=v(1:3)/gamma;
R(3,:)=R(1,:).*R(2,:);
T=zeros(3,1);
T(1)=v(8,1)/gamma;
T(2)=v(4,1)/gamma;
%% Ortogonalitatea lui R
% Se doreste impunerea ortogonalitatii matrici de rotatie astfel incat
% R*Rt=I3. In acest scop se utilizeaza metoda SVD in vederea
descompunerii
% matrici de rotatie R in U1,D1,V1. Matricea Di se inlocuieste cu I3.
I3=eye(3);
[U1 D1 V1]=svd(R);
R=U1*I3*V1';
%% Semnul lui gamma
% Se doreste determinarea semnului necunoscut al factorului de scalare
% gamma in vederea estimarii definitive a parametrilor.
r11=R(1,1);
r12=R(1,2);
r13=R(1,3);
r21=R(2,1);
r22=R(2,2);
r23=R(2,3);
```

```
x=MatPct2D(1,1);
y=MatPct2D(1,2);
X=MatPct3D(1,1);
Y=MatPct3D(1,2);
Z=MatPct3D(1,3);
semnx=x*(r11*X+r12*Y+r13*Z+T(1));
semny=y*(r21*x+r22*Y+r23*Z+T(2));
if semnx>0
    smxr=-1;
else
    smxr=1;
end
if semnx>0
    smxt=-sign(T(1));
else
    smxt=sign(T(1));
end
if semny>0
    smyr=-1;
else
    smyr=1;
end
if semny>0
    smyt=-sign(T(2));
else
    smyt=sign(T(2));
end
%% Matricea R si T partial
% Se reface matricea de rotatie si cele doua componente ale vectorului
de
% translatie cu ajutorul semnului lui gama cunoscut
R(1,:)=R(1,:).*smxr;
R(2,:)=R(2,:).*smyr;
T(1)=smxt*T(1);
T(2)=smyt*T(2);
%% Determinarea Tz si f
% Se doreste determinarea parametrilor ramasi si anume ultimului termen
al
% vectorului de translatie Tz si determinarea distantei focale
dealungul
% axei x in pixeli fx, folosind metoda celor mai mici patrate (least
% squares solution).
B=[xim R(1,1)*Xw+R(1,2)*Yw+R(1,3)*Zw+T(1)];
b=[-xim.*(R(3,1)*Xw+R(3,2)*Yw+R(3,3)*Zw)];
Bpsi=pinv(B);
C=Bpsi*b;
T(3)=C(1);
fx=C(2);
%% Parametri rezultati
% Se doreste exprimarea explicita a parametrilor extrinseci si
intrinseci
MatRotatie=R;
MatTranslatie=T;
DisFocala=fx;
```

Anexa 3: Codul sursă, în Matlab, pentru metoda Faugeras de calibrare a camerelor video.

```
%##################Cristian Pop – Student Drd.################
%                CALIBRAREA UNEI CAMERE VIDEO
%                PE BAZA METODEI LUI FAUGERAS
%
% Date de intrare: Doua fisier text ce contine coordonatele unor puncte
din
%                          realitate raportate la un sistem de referinta
global fix
%                                si repectiv coordonatele in pixeli ale
corespondentelor
%                         lor in planul imagine.
% Date de iesire: Parametri intrinseci si extrinseci
%#########################################################
%%
function [MR, MT, MC, Q, F, O, M3D2D, Er]= CalibrareMS5(~, ~)
% Citeste cele doua fisiere de coordonate
[m3D]=textread('CoordonateA8Pct3D.txt');
[nR3D, nC3D]=size(m3D);
if(nC3D~=3)
    error('Matricea 3D nu detine trei coloane.');
end
Xw = m3D(:,1);
Yw = m3D(:,2);
Zw = m3D(:,3);
[m2D]=textread('CoordonateA8Pct2D.txt');
[nR2D, nC2D]=size(m2D);
if(nC2D~=2)
    error('Matricea 2D nu detine doua coloane.');
end
xim=m2D(:,1);
yim=m2D(:,2);
if(nR3D~=nR2D)
    error('Verifica ca numarul punctelor 2d si 3d sa fie acelasi.');
end
% Se va nota matricea de calibrare cu M;
% ecuatia lineara va fi de forma AV=0;
% unde A este o matrice de tipul 2nx12
% iar V este un vector format din 12 necunoscute.
%%
o = ones(size(xim));
z = zeros(size(xim));
RandImparA  = [ Xw Yw Zw o z z z z -xim.*Xw -xim.*Yw -xim.*Zw -xim ];
RandParA = [ z z z z Xw Yw Zw o -yim.*Xw -yim.*Yw -yim.*Zw -yim ];
A=[RandImparA; RandParA]
[U, S, V] = svd(A,0);
m = V(:,end);
M = reshape(m,4,3)';
MC=M;
q1=M(1,1:3);
q2=M(2,1:3);
q3=M(3,1:3);
```

```
q4=M(1:3,4);
gamma_abs=sqrt(M(3,1)^2 + M(3,2)^2 + M(3,3)^2);
M = M/gamma_abs;
inFata=1;
if inFata
    s = sign(M(3,4));
else
    s = -sign(M(3,4));
end
%% Calcularea lui T_z
T(3) = s*M(3,4);
R = zeros(3,3);
R(3,:)=s*M(3,1:3);
% se calculeaza qi;
q1 = M(1,1:3)';
q2 = M(2,1:3)';
q3 = M(3,1:3)';
q4 = M(1:3,4);
% Calcularea centrul proiectiei ox si oy
ox = q1'*q3;
oy = q2'*q3;
% Calcularea distantei focale dealungul lui xim si yim respectiv fx si
fy
fx=sqrt(q1'*q1-ox^2);
fy=sqrt( q2'*q2 - oy^2 );
%%
R(1,:) = s*(ox*M(3,1:3) - M(1,1:3) ) / fx;
R(2,:) = s*(oy*M(3,1:3) - M(2,1:3) ) / fy;
T(1) = s*(ox*M(3,4) - M(1,4) ) / fx;
T(2) = s*(oy*M(3,4) - M(2,4) ) / fy;
T = T';
MT=T;
%%
[U,D,V] = svd(R);
R = U*V';
MR=R;
nou2D=zeros(nR3D,2);
for i=1:1:nR3D
    num_x=M(1,1)*m3D(i,1) + M(1,2)*m3D(i,2) + M(1,3)*m3D(i,3) + M(1,4);
    num_y=M(2,1)*m3D(i,1) + M(2,2)*m3D(i,2) + M(2,3)*m3D(i,3) + M(2,4);
    den=M(3,1)*m3D(i,1) + M(3,2)*m3D(i,2) + M(3,3)*m3D(i,3) + M(3,4);
    nou2D(i,1)=num_x/den;
    nou2D(i,2)=num_y/den;
end
Q=[q1;q2;q3;q4];
F=[fx fy];
O=[ox oy];
M3D2D=nou2D;
errorDiff=M3D2D-m2D;
er_x=mean(errorDiff(:,1));
er_y=mean(errorDiff(:,2));
Er=[er_x er_y];
```

Anexa 4: Programul în Matlab pentru aplicaţia de inspectare a suprafeţelor

```
function varargout = atreiaC(varargin)
% ATREIAC M-file for atreiaC.fig
%      ATREIAC, by itself, creates a new ATREIAC or raises the existing
%      singleton*.
%      H = ATREIAC returns the handle to a new ATREIAC or the handle to
%      the existing singleton*.
%          ATREIAC('CALLBACK',hObject,eventData,handles,...) calls the
local
%         function named CALLBACK in ATREIAC.M with the given input
arguments.
%       ATREIAC('Property','Value',...) creates a new ATREIAC or raises
the
%        existing singleton*.  Starting from the left, property value
pairs are
%      applied to the GUI before atreiaC_OpeningFcn gets called.  An
%         unrecognized property name or invalid value makes property
application
%      stop.  All inputs are passed to atreiaC_OpeningFcn via varargin.
%      *See GUI Options on GUIDE's Tools menu.  Choose "GUI allows only
one
%      instance to run (singleton)".
% See also: GUIDE, GUIDATA, GUIHANDLES
% Edit the above text to modify the response to help atreiaC
% Last Modified by GUIDE v2.5 04-Feb-2010 23:53:27
% Begin initialization code - DO NOT EDIT
gui_Singleton = 1;
gui_State = struct('gui_Name',       mfilename, ...
                   'gui_Singleton',  gui_Singleton, ...
                   'gui_OpeningFcn', @atreiaC_OpeningFcn, ...
                   'gui_OutputFcn',  @atreiaC_OutputFcn, ...
                   'gui_LayoutFcn',  [] , ...
                   'gui_Callback',   []);
if nargin && ischar(varargin{1})
    gui_State.gui_Callback = str2func(varargin{1});
end
if nargout
    [varargout{1:nargout}] = gui_mainfcn(gui_State, varargin{:});
else
    gui_mainfcn(gui_State, varargin{:});
end
% End initialization code - DO NOT EDIT
% --- Executes just before atreiaC is made visible.
function atreiaC_OpeningFcn(hObject, eventdata, handles, varargin)
% This function has no output args, see OutputFcn.
% hObject    handle to figure
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
% varargin   command line arguments to atreiaC (see VARARGIN)
% Choose default command line output for atreiaC
handles.output = hObject;
% Update handles structure
```

```
guidata(hObject, handles);
% UIWAIT makes atreiaC wait for user response (see UIRESUME)
% uiwait(handles.figure1);
% --- Outputs from this function are returned to the command line.
function varargout = atreiaC_OutputFcn(hObject, eventdata, handles)
% varargout  cell array for returning output args (see VARARGOUT);
% hObject    handle to figure
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
% Get default command line output from handles structure
varargout{1} = handles.output;
% --- Executes on button press in incarca_button1.
function incarca_button1_Callback(hObject, eventdata, handles)
% hObject    handle to incarca_button1 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
[nume,cale]=uigetfile('*.jpg');
I1=imread(strcat(char(cale),char(nume)));
subplot(3,3,[2 3]);
himage=I1;
iptsetpref('ImshowAxesVisible','on')
Imshow(himage,'DisplayRange',[]);
hold on;
title('prima imagine');
% --- Executes on button press in incarca_button2.
function incarca_button2_Callback(hObject, eventdata, handles)
% hObject    handle to incarca_button2 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
[nume,cale]=uigetfile('*.jpg');
I2=imread(strcat(char(cale),char(nume)));
subplot(3,3,[5 6]);
himage=I2;
iptsetpref('ImshowAxesVisible','on')
Imshow(himage,'DisplayRange',[]);
title('a doua imagine');
% --- Executes on button press in Etalonare_button.
function Etalonare_button_Callback(hObject, eventdata, handles)
% hObject    handle to Etalonare_button (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
etalonare=get(handles.Edit_etalonare,'String');
val_etalonare=str2double(etalonare);
[dr,st]=ginput(2);
        dist_pixeli=sqrt((dr(1)-dr(2))^2+(st(1)-st(2))^2);
        marime_pixeli=dist_pixeli/val_etalonare;
        handles.Etalonare_push=marime_pixeli;
guidata(hObject,handles)
function Edit_etalonare_Callback(hObject, eventdata, handles)
% hObject    handle to Edit_etalonare (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
% Hints: get(hObject,'String') returns contents of Edit_etalonare as
text
```

```
%               str2double(get(hObject,'String'))  returns  contents  of
Edit_etalonare as a double
% --- Executes during object creation, after setting all properties.
function Edit_etalonare_CreateFcn(hObject, eventdata, handles)
% hObject    handle to Edit_etalonare (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles     empty - handles not created until after all CreateFcns
called
% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if        ispc        &&        isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end
% --- Executes on button press in crop_button.
function crop_button_Callback(hObject, eventdata, handles)
% hObject    handle to crop_button (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
SelecteazaPunct(hObject, eventdata, handles);
function SelecteazaPunct(hObject, eventdata, handles)
% hObject    handle to select (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
[p1,p2]=ginput(1);
I3 = imread('poza2.jpg');
x0=p1;
y0=p2-p2;
xl=150*handles.Etalonare_push;
I4 = imcrop(I3,[x0 y0 xl 2448]);
subplot(3,3,[5 6]);
himage=I4;
iptsetpref('ImshowAxesVisible','on')
Imshow(himage,'DisplayRange',[]);
hold on;
title('a doua imagine taiata');
save('mySave.mat', 'I4')
% --- Executes on button press in lipeste_button.
function lipeste_button_Callback(hObject, eventdata, handles)
% hObject    handle to lipeste_button (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
StitchImages(hObject, eventdata, handles);
function StitchImages(hObject, eventdata, handles)
% hObject    handle to btnStitch (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
global stitchedImage;     %  Stitched_Image_Click_Event  needs  to  use
this array.
I5=imread('poza1.jpg');
load('mySave.mat', 'I4');
I6=I4;
stitchedImage = [I5 I6];
subplot(3,3,[8 9]);
I7=stitchedImage;
```

```
iptsetpref('ImshowAxesVisible','on')
Imshow(I7,'DisplayRange',[]);
hold on;
title('Imaginea rezultata');
save('mySave.mat', 'I7');
% --- Executes on button press in salveaza_button.
function salveaza_button_Callback(hObject, eventdata, handles)
% hObject    handle to salveaza_button (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
% --- Executes on button press in ajustare_button.
function ajustare_button_Callback(hObject, eventdata, handles)
% hObject    handle to ajustare_button (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
load('mySave.mat', 'I7');
I=I7(:,:,1);
handles.im=I;
imshow(handles.im, 'DisplayRange',[])
I=handles.im;
imcontrast(gcf);
guidata(hObject,handles)
% --- Executes on slider movement.
function slider_tresh_Callback(hObject, eventdata, handles)
% hObject    handle to slider_tresh (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
% Hints: get(hObject,'Value') returns position of slider
%         get(hObject,'Min') and get(hObject,'Max') to determine range
of slider
nivel=get(handles.slider_tresh,'Value');
nivel=round(nivel);
bw1=handles.im<nivel;
imshow(bw1)
str=sprintf('%3d',round(nivel));
set(handles.edit_tresh,'String',nivel);
handles.bw=bw1;
%handles.im=bw1;
guidata(hObject,handles)
% --- Executes during object creation, after setting all properties.
function slider_tresh_CreateFcn(hObject, eventdata, handles)
% hObject    handle to slider_tresh (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles     empty - handles not created until after all CreateFcns
called
% Hint: slider controls usually have a light gray background.
if                              isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor',[.9 .9 .9]);
end
function edit_tresh_Callback(hObject, eventdata, handles)
% hObject    handle to edit_tresh (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
% Hints: get(hObject,'String') returns contents of edit_tresh as text
```

```
%               str2double(get(hObject,'String'))  returns  contents  of
edit_tresh as a double
set(handles.edit_thresh,'Value',nivel);
guidata(hObject,handles)
% --- Executes during object creation, after setting all properties.
function edit_tresh_CreateFcn(hObject, eventdata, handles)
% hObject    handle to edit_tresh (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles     empty - handles not created until after all CreateFcns
called
% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if        ispc        &&        isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end
function edit_lung_Callback(hObject, eventdata, handles)
% hObject    handle to edit_lung (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
% Hints: get(hObject,'String') returns contents of edit_lung as text
%               str2double(get(hObject,'String'))  returns  contents  of
edit_lung as a double
%lungime=handles.edit_lung;
%set(handles.edit_lung,'Value',lungime)
%guidata(hObject,handles)
% --- Executes during object creation, after setting all properties.
function edit_lung_CreateFcn(hObject, eventdata, handles)
% hObject    handle to edit_lung (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles     empty - handles not created until after all CreateFcns
called
% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if        ispc        &&        isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end
function edit_unghi_Callback(hObject, eventdata, handles)
% hObject    handle to edit_unghi (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
% Hints: get(hObject,'String') returns contents of edit_unghi as text
%               str2double(get(hObject,'String'))  returns  contents  of
edit_unghi as a double
% --- Executes during object creation, after setting all properties.
function edit_unghi_CreateFcn(hObject, eventdata, handles)
% hObject    handle to edit_unghi (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles     empty - handles not created until after all CreateFcns
called
% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if        ispc        &&        isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
```

```
    set(hObject,'BackgroundColor','white');
end
% --- Executes on button press in Calibration_push.
function Calibration_push_Callback(hObject, eventdata, handles)
% hObject    handle to Calibration_push (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
etalon=get(handles.Edit_etalon,'String');
val_etalon=str2double(etalon);
[re,ce]=ginput(2);
        dist_pixeli=sqrt((re(1)-re(2))^2+(ce(1)-ce(2))^2);
        scalare=dist_pixeli/val_etalon;
        handles.Calibration_push=scalare;
guidata(hObject,handles)
function Edit_etalon_Callback(hObject, eventdata, handles)
% hObject    handle to Edit_etalon (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
% Hints: get(hObject,'String') returns contents of Edit_etalon as text
%               str2double(get(hObject,'String'))  returns  contents  of
Edit_etalon as a double
% --- Executes during object creation, after setting all properties.
function Edit_etalon_CreateFcn(hObject, eventdata, handles)
% hObject    handle to Edit_etalon (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles     empty - handles not created until after all CreateFcns
called
% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if        ispc        &&        isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end
% --- Executes on selection change in popup_tip.
function popup_tip_Callback(hObject, eventdata, handles)
% hObject    handle to popup_tip (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
% Hints: contents = get(hObject,'String') returns popup_tip contents as
cell array
%           contents{get(hObject,'Value')} returns selected item from
popup_tip
val = get(hObject,'Value');
bw=handles.bw;
switch val
    case 1
       [x1,y1]=ginput(2);
       y=-y1;x=ceil(x1);
       t1=handles.bw;
       m=(y(2)-y(1))/(x(2)-x(1));
        nr_p=x(2)-x(1);
        if nr_p<0
            for i=1:-1:nr_p
                Y=abs(y(1)+m*i);
                coord_x=(x(1)+i);
```

```
                coord_y=ceil(Y);
                t1(coord_y,coord_x)=1;
                coord_y=floor(Y);
                t1(coord_y,coord_x)=1;
            end
        else
            for i=1:nr_p
                Y=abs(y(1)+m*i);
                coord_x=(x(1)+i);
                coord_y=ceil(Y);
                t1(coord_y,coord_x)=1;
                coord_y=floor(abs(Y));
                t1(coord_y,coord_x)=1;
            end
        end
           imshow(t1)
           lungime=sqrt((x(1)-x(2))^2+(y(1)-y(2))^2);
           lungime=lungime/handles.Calibration_push;
           str=sprintf('%3d',lungime);
           set(handles.edit_lung,'String',str)
           unghi=atand(m);
           str=sprintf('%3d',unghi);
           set(handles.edit_unghi,'String',str)
    case 2
        [x1,y1]=ginput;
        y=-y1;x=ceil(x1);
        t1=handles.bw;
        m=(y(2)-y(1))/(x(2)-x(1));
        n=length(x);
        lungime=0;
        for i=1:2:n-1
            lungime=lungime+sqrt((x(i)-x(i+1))^2+(y(i)-y(i+1))^2);

            m0=(y(i+1)-y(i))/(x(i+1)-x(i));
        nr_p=x(i+1)-x(i);
        if nr_p<0
            for j=1:-1:nr_p
                Y=abs(y(i)+m0*j);
                coord_x=(x(i)+j);
                coord_y=ceil(Y);
                t1(coord_y,coord_x)=1;
                coord_y=floor(Y);
                t1(coord_y,coord_x)=1;
            end
        else
            for j=1:nr_p
                Y=abs(y(i)+m0*j);
                coord_x=(x(i)+j);
                coord_y=ceil(Y);
                t1(coord_y,coord_x)=1;
                coord_y=floor(abs(Y));
                t1(coord_y,coord_x)=1;
            end
        end
           imshow(t1)
```

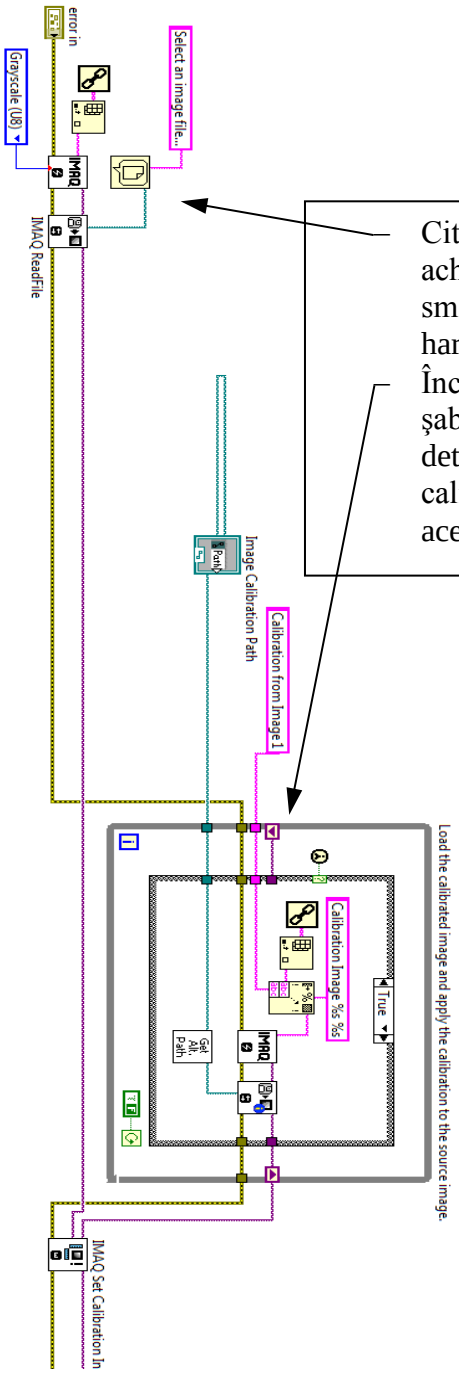
```
        end
        lungime=lungime/handles.Calibration_push;
           str=sprintf('%1d',lungime);
           set(handles.edit_lung,'String',str)
           unghi=atand(m);
           str=sprintf('%1d',unghi);
           set(handles.edit_unghi,'String',str)
    case 3
        [x1,y1]=ginput;
        y=-y1;x=ceil(x1);
        t1=handles.bw;
        n=length(x);
        lungime=0;
        for i=1:n-1
           lungime=lungime+sqrt((x(i)-x(i+1))^2+(y(i)-y(i+1))^2);

            m0=(y(i+1)-y(i))/(x(i+1)-x(i));
        nr_p=x(i+1)-x(i);
        if nr_p<0
            for j=1:-1:nr_p
                Y=abs(y(i)+m0*j);
                coord_x=(x(i)+j);
                coord_y=ceil(Y);
                t1(coord_y,coord_x)=1;
                coord_y=floor(Y);
                t1(coord_y,coord_x)=1;
            end
        else
            for j=1:nr_p
                Y=abs(y(i)+m0*j);
                coord_x=(x(i)+j);
                coord_y=ceil(Y);
                t1(coord_y,coord_x)=1;
                coord_y=floor(abs(Y));
                t1(coord_y,coord_x)=1;
            end
        end
           imshow(t1)

        end
        lungime=lungime/handles.Calibration_push;
           str=sprintf('%1d',lungime);
           set(handles.edit_lung,'String',str)
           set(handles.edit_unghi,'String','fisura este curbilinie')
end
guidata(hObject,handles)
% --- Executes during object creation, after setting all properties.
function popup_tip_CreateFcn(hObject, eventdata, handles)
% hObject    handle to popup_tip (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles     empty - handles not created until after all CreateFcns
called
% Hint: popupmenu controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
```

```
if        ispc        &&        isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end
% --- Executes on selection change in popup_corect.
function popup_corect_Callback(hObject, eventdata, handles)
% hObject    handle to popup_corect (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
% Hints: contents = get(hObject,'String') returns popup_corect contents
as cell array
%           contents{get(hObject,'Value')} returns selected item from
popup_corect
val = get(hObject,'Value');
bw=handles.bw;
handles.im=bw;
switch val
    case 1
        bw1=bwmorph(bw,'open');
    case 2
        bw1=bwmorph(bw,'close');
end
imshow(bw1)
handles.bw=bw1;
guidata(hObject,handles)
% --- Executes during object creation, after setting all properties.
function popup_corect_CreateFcn(hObject, eventdata, handles)
% hObject    handle to popup_corect (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles     empty - handles not created until after all CreateFcns
called
% Hint: popupmenu controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if        ispc        &&        isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end
% --- Executes on button press in push_crop.
function push_crop_Callback(hObject, eventdata, handles)
% hObject    handle to push_crop (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
[In,rect]=imcrop(handles.bw);
q1=rect(1);q2=rect(2);q3=rect(3);q4=rect(4);
bw2=handles.bw;
bw2(1:q2,:)=0;bw2(:,1:q1)=0;bw2(q2+q4:end,:)=0;bw2(:,q1+q3:end)=0;
imshow(bw2)
% --- Executes on button press in reincarca_button.
function reincarca_button_Callback(hObject, eventdata, handles)
% hObject    handle to reincarca_button (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
I0=handles.im;
imshow(I0)
handles.bw=I0;
```

```
guidata(hObject,handles)
```

Anexa 5: Diagrama programului implementat în Labview, de control al unui robot cartezian, pe bază de imagini.

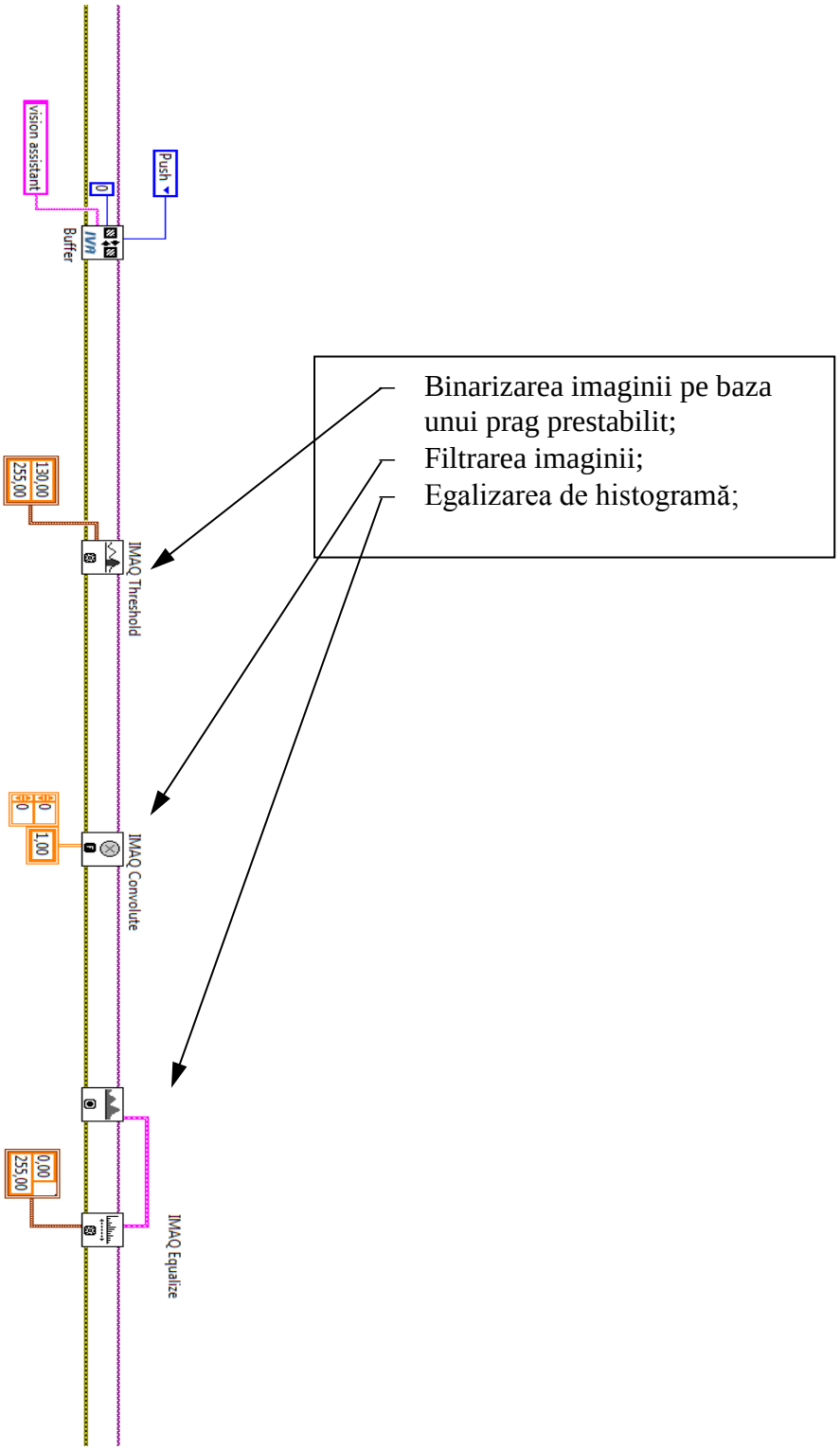
Pagina 1/5



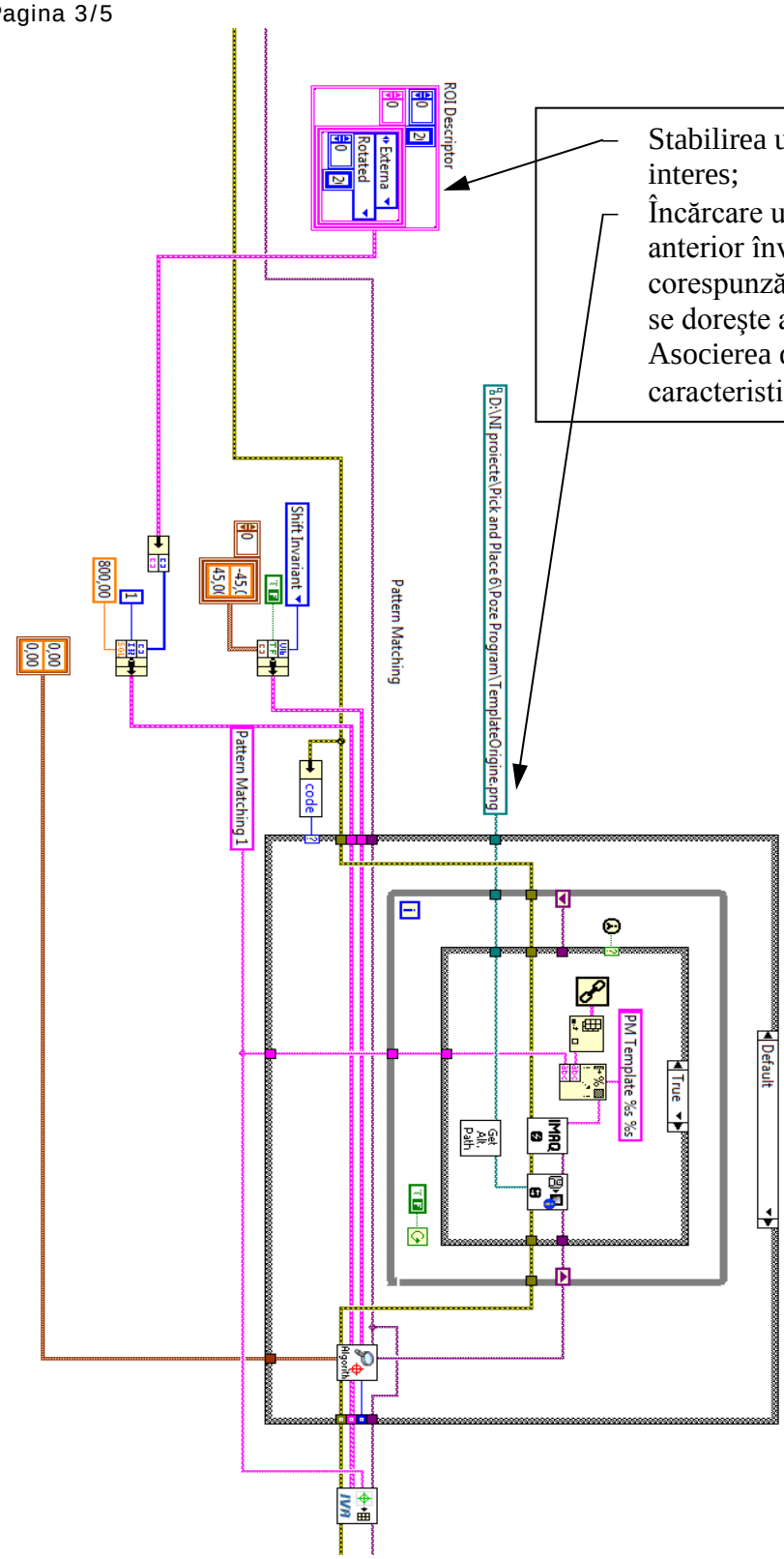
Citire şi încărcare imagine achiziţionată de camera smart, stocată pe unitatea hard a PC-ului;

Încărcare imagine cu şablonul de calibrare, determinare parametri de calibrare şi memorarea acestora;

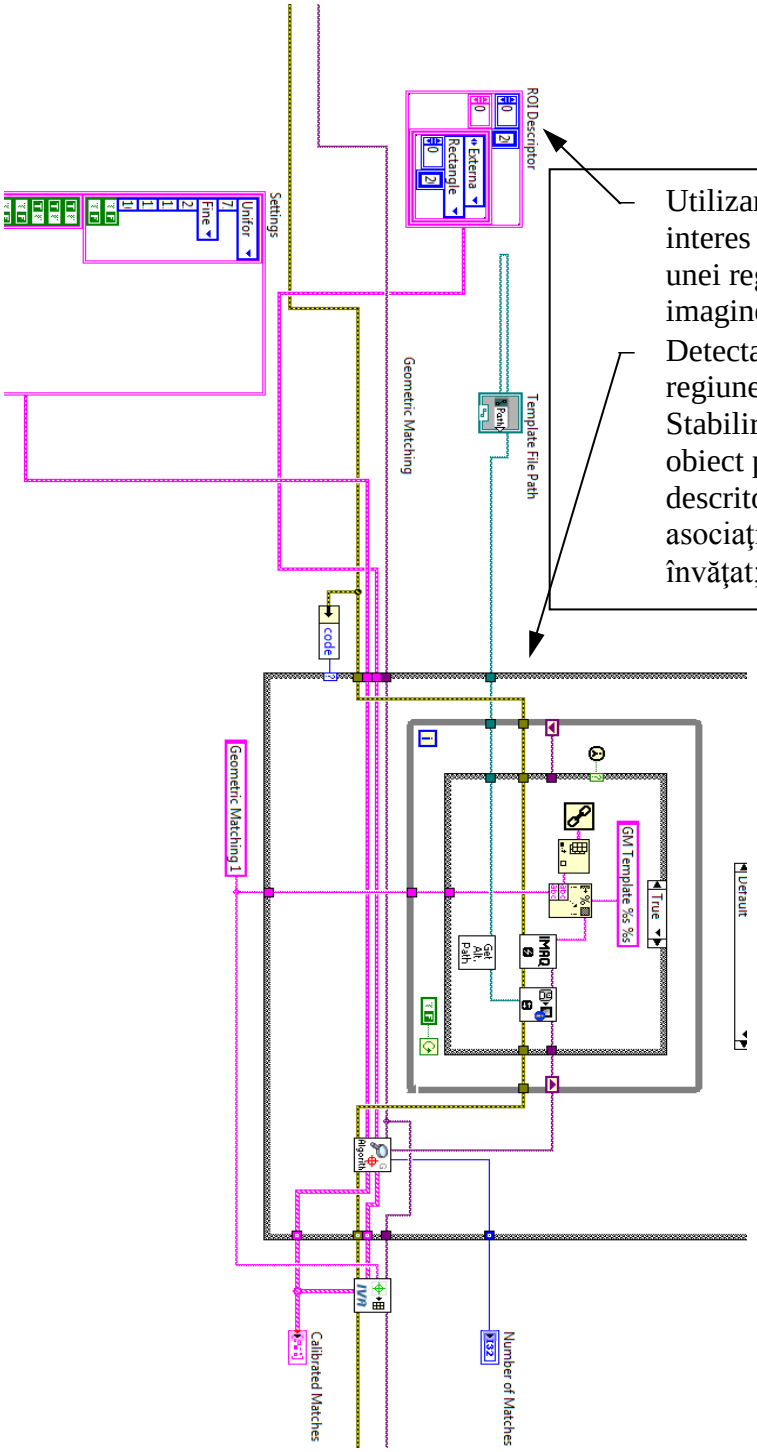
Pagina 2/5


vision assistant
Push
Buffer
130,00
255,00
IMAQ Threshold
0
0
1,00
IMAQ Convolute
0,00
255,00
IMAQ Equalize
Binarizarea imaginii pe baza unui prag prestabilit;
Filtrarea imaginii;
Egalizarea de histogramă;

Pagina 3/5


ROI Descriptor
Rotated
Externa
Stabilirea unei regiuni de interes;
Încărcare unui şablon, anterior învăţat, corespunzător obiectului ce se doreşte a fi detectat.
Asocierea diverselor caracteristici de formă;
D:\NI proiecte\Pick and Place 6\Poze Program\TemplateOrigine.png
Pattern Matching
Shift Invariant
800,00
0,00
0,00
Pattern Matching 1
code
PM Template %s %s
True
Default
Get Alt. Path
IMAQ

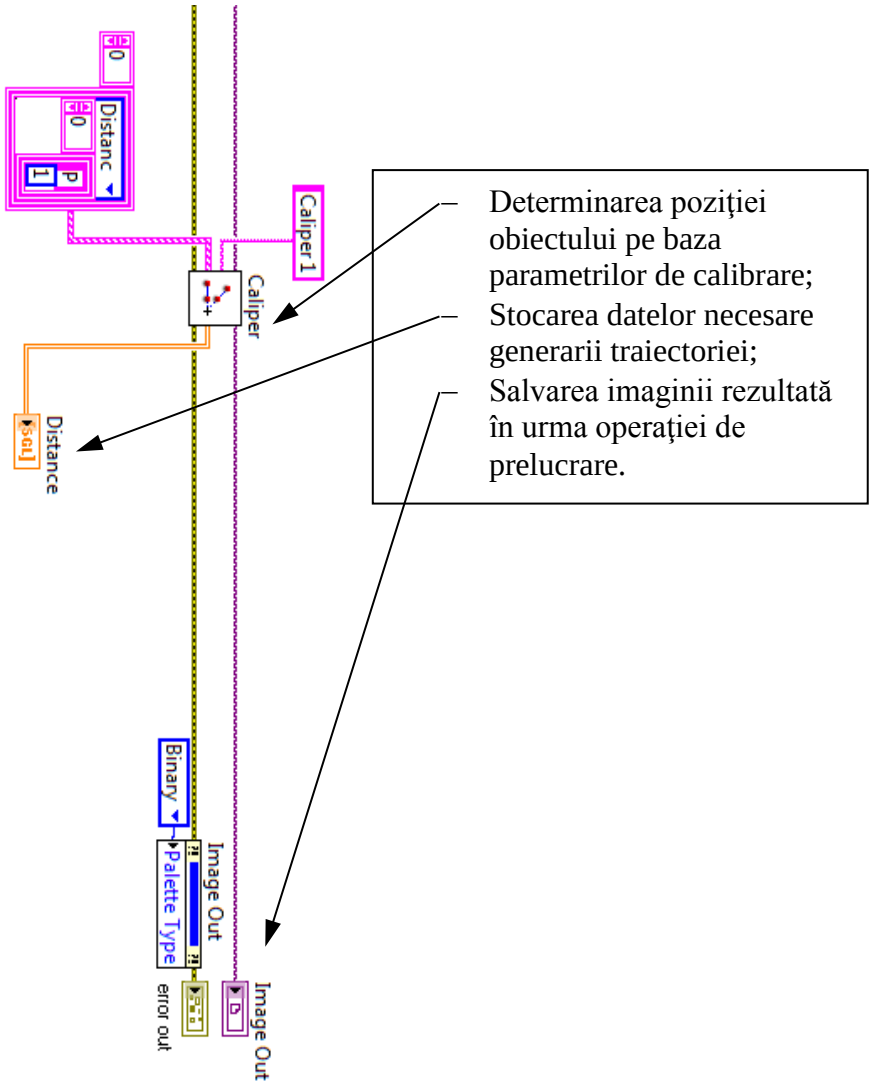
Pagina 4/5



Utilizarea zonei de interes pentru analizarea unei regiuni din imagine;

Detectarea obiectului în regiunea respectivă. Stabilirea tipului de obiect pe baza unor descritori de contur asociaţi şablonului învăţat;

Pagina 5/5


0
Distanc
0
P
1
Caliper 1
Caliper
Distance
Binary
Palette Type
Image Out
error out
Image Out
Determinarea poziţiei obiectului pe baza parametrilor de calibrare;
Stocarea datelor necesare generarii traiectoriei;
Salvarea imaginii rezultată în urma operaţiei de prelucrare.

Anexa 6: Programul pentru generarea fişierului „Traiectorie.txt” necesar conducerii robotului

```
function varargout = Traiectorie(varargin)
% TRAIECTORII M-file for traiectorii.fig
% TRAIECTORII, by itself, creates a new TRAIECTORII or raises the
existing
% singleton*.
% H = TRAIECTORII returns the handle to a new TRAIECTORII or the handle
to
% the existing singleton*.
% TRAIECTORII('CALLBACK',hObject,eventData,handles,...) calls the local
% function named CALLBACK in TRAIECTORII.M with the given input
arguments.
% TRAIECTORII('Property','Value',...) creates a new TRAIECTORII or
raises the
% existing singleton*.  Starting from the left, property value pairs
are
% applied to the GUI before traiectorii_OpeningFcn gets called.  An
% unrecognized property name or invalid value makes property
application
% stop.  All inputs are passed to traiectorii_OpeningFcn via varargin.
% *See GUI Options on GUIDE's Tools menu.  Choose "GUI allows only one
% instance to run (singleton)".
% See also: GUIDE, GUIDATA, GUIHANDLES
% Edit the above text to modify the response to help traiectorii
% Last Modified by GUIDE v2.5 28-Apr-2011 03:24:12
% Begin initialization code - DO NOT EDIT
gui_Singleton = 1;
gui_State = struct('gui_Name',       mfilename, ...
                   'gui_Singleton',  gui_Singleton, ...
                   'gui_OpeningFcn', @traiectorii_OpeningFcn, ...
                   'gui_OutputFcn',  @traiectorii_OutputFcn, ...
                   'gui_LayoutFcn',  [] , ...
                   'gui_Callback',   []);
if nargin && ischar(varargin{1})
    gui_State.gui_Callback = str2func(varargin{1});
end
if nargout
    [varargout{1:nargout}] = gui_mainfcn(gui_State, varargin{:});
else
    gui_mainfcn(gui_State, varargin{:});
end
% End initialization code - DO NOT EDIT
% --- Executes just before traiectorii is made visible.
function traiectorii_OpeningFcn(hObject, eventdata, handles, varargin)
% This function has no output args, see OutputFcn.
% hObject    handle to figure
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
% varargin   command line arguments to traiectorii (see VARARGIN)
% Choose default command line output for traiectorii
handles.output = hObject;
% Update handles structure
```

```
guidata(hObject, handles);
% UIWAIT makes traiectorii wait for user response (see UIRESUME)
% uiwait(handles.figure1);
% --- Outputs from this function are returned to the command line.
function varargout = traiectorii_OutputFcn(hObject, eventdata, handles)
% varargout  cell array for returning output args (see VARARGOUT);
% hObject    handle to figure
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
% Get default command line output from handles structure
varargout{1} = handles.output;
% --- Executes on button press in pushbutton1.
function pushbutton1_Callback(hObject, eventdata, handles)
% hObject    handle to pushbutton1 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
valx=str2num(get(handles.editx,'String'));
valy=str2num(get(handles.edity,'String'));
f=fopen('Traiectorie.txt','w');
N=10000;
for a=1:1:N
    x=0;
    y=0;
    z=a/N*(-150);
    fprintf(f,'%6f %6f %6f\n',x,y,z);
end
for b=1:1:N
    x=b/N*(valx);
    y=b/N*(valy);
    z=-150;
    fprintf(f,'%6f %6f %6f\n',x,y,z);
end
for c=1:1:N
    x=valx;
    y=valy;
    z=-150+(c/N*150);
    fprintf(f,'%6f %6f %6f\n',x,y,z);
end
for d=1:1:N
    x=valx*(d/N*N/d);
    y=valy*(d/N*N/d);
    z=0;
    fprintf(f,'%6f %6f %6f\n',x,y,z);
end
for e=1:1:N
    x=valx;
    y=valy;
    z=e/N*(-150);
    fprintf(f,'%6f %6f %6f\n',x,y,z);
end
for g=1:1:N
    x=valx-(g/N*valx);
    y=valy-(g/N*valy);
    z=-150;
    fprintf(f,'%6f %6f %6f\n',x,y,z);
```

```
end
for h=1:1:N
    x=0;
    y=0;
    z=-150+(h/N*150);
    fprintf(f,'%6f %6f %6f\n',x,y,z);
end
fclose(f);
function editx_Callback(hObject, eventdata, handles)
% hObject    handle to editx (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
% Hints: get(hObject,'String') returns contents of editx as text
%        str2double(get(hObject,'String')) returns contents of editx as
a double
% --- Executes during object creation, after setting all properties.
function editx_CreateFcn(hObject, eventdata, handles)
% hObject    handle to editx (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns
called
% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end
function edity_Callback(hObject, eventdata, handles)
% hObject    handle to edity (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
% Hints: get(hObject,'String') returns contents of edity as text
%        str2double(get(hObject,'String')) returns contents of edity as
a double
% --- Executes during object creation, after setting all properties.
function edity_CreateFcn(hObject, eventdata, handles)
% hObject    handle to edity (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns
called
% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end
```

Anexa 7: Program, în Matlab, pentru detectarea obiectului pe bază de culoare în spaţiul HSV

```
function [cubu]=APNR1PrelucrareImagine(piesa)
% Prelucrare imagine
% Citeste imaginea
[rgbImage storedColorMap] = imread(piesa);
% Determina numarul de pixeli pe verticala si orizontala si numarul de
% plane de culoare.
[Randuri Coloane NumarBenziCuloare] = size(rgbImage);
% Convert RGB image to HSV
hsvImage = rgb2hsv(rgbImage);
% figure (12), imshow(hsvImage);
PragMinH=0.180;
PragMaxH=0.250;
PragMinS=0.400;
PragMaxS=0.740;
PragMinV=0.200;
PragMaxV=0.460;
[h,w] = size(rgbImage(:,:,1));
binImage = zeros(h,w);
for i=1:h
   for j=1:w
       binImage(i,j) = hsvImage(i,j,1)>PragMinH &&
hsvImage(i,j,1)<PragMaxH...
                         && hsvImage(i,j,2)>PragMinS &&
hsvImage(i,j,2)<PragMaxS...
                         && hsvImage(i,j,3)>PragMinV &&
hsvImage(i,j,3)<PragMaxV;
   end
end
% Operatii morfologice
% Elimina obiectele ce sunt mai mici de 100 de pixeli
MarimeObiectAcceptat = 100;
ObiectVerde = bwareaopen(binImage, MarimeObiectAcceptat);
% Inchide conturul obiectelor;
se = strel('square',6);
bw = imclose(ObiectVerde,se);
% Aplica operatia morfologica de umplere de goluri;
bw1 = imfill(bw,'holes');
% Aplica operatia morfologica de erodare a obiectelor cu x pixeli;
nrpixeli=strel('square',6);
bw2=imerode(bw1,nrpixeli);
% Aplica operatia morfologica de dilatare a obiectelor cu x pixeli;
bw3=imdilate(bw2,nrpixeli);
% Aplica operatia morfologica de inchidere a obiectelor cu x pixeli;
se2 = strel('line', 21, 90);
bw4a = imclose(bw3,se2);
se3 = strel('line', 21, 0);
bw4b = imclose(bw3,se3);
figure, imshow(bw4b)
% imwrite(bw4b, 'Obiect.png');
% Determina conturul obiectului;
bw5=bwperim(bw4b);
```

```
figure, imshow(bw5)
% imwrite(bw5, 'ConturObiect.png');
%etichetare obiecte din imagine;
aux=bwlabel(bw5);
%determinare indicatori obiect;
stat=regionprops(aux,'Area','BoundingBox');
%Conturul piesei pleaca din pct de coordonate r=stat.BoundingBox(1),
%c=stat.BoundingBox(2)si are latimea pe r stat.BoundingBox(3) si pe c
%stat.BoundingBox(4);
Iprelucrat=zeros(size(bw5));
c0=floor(stat.BoundingBox(1)+.5);
r0=floor(stat.BoundingBox(2)+.5);
Iprelucrat(r0,c0)=1;
latime=floor(stat.BoundingBox(3))-1;lungime=floor(stat.BoundingBox(4))-
1;
%trasare contur
for i=1:latime
    Iprelucrat(r0,c0+i)=1;
    Iprelucrat(r0+lungime,c0+i)=1;
end
for i=1:lungime
    Iprelucrat(r0+i,c0)=1;
    Iprelucrat(r0+i,c0+latime)=1;
end
aux1=bwlabel(Iprelucrat);
stat1=regionprops(aux1,'Area','BoundingBox');
figure, imshow(Iprelucrat)
% imwrite(Iprelucrat, 'ConturPrelucrat.png');
% Umple conturul obiectului;
cubu=imfill(Iprelucrat);
hold on;
figure, imshow(cubu);
CubBinar=cubu;
% Salveaza imaginea prelucrata;
imwrite(cubu, 'CubBinar.png');
```

Anexa 8: Programul din controlerul robotului pentru aplicaţia cu paralelipipede

```
  PROGRAM   CRIST
                ********************
   3467: LABEL     1
   3468: SET       A = 0
   3469: WAIT      A <> 0
   3470: SPEED     20
   3471: MOVE      CIM[150]
   3472: GOSUB     OGRIP
   3473: MOVED     POP[10]
   3474: SPEED     10
   3475: SET       B=-A - 59755
   3476: SETPVC    POP[2] Z  B
   3477: MOVED     POP[2]
   3478: GOSUB     CGRIP
   3479: SPEED     20
   3480: MOVED     POP[10]
   3481: MOVED     CIM[12]
   3482: SPEED     10
   3483: SET       C=-A - 82000
   3484: SETPVC    CIM[2] Z  C
   3485: MOVED     CIM[2]
   3486: GOSUB     OGRIP
   3487: MOVED     CIM[12]
   3488: GOSUB     CGRIP
   3489: MOVED     CIM[150]
   3490: GOTO      1
   3491: END
(END)
```

Anexa 9: Program pentru recunoaşterea tipului de rulment pe bază de şablon

```
function[TipRulment]=ComparaRulmentCuSablon(contur)
RulDetectat=contur;
%comparare rulmenti
sablon1=imread('sablon1.tif');
ValPerimSablon1=1076;
sablon2=imread('sablon2.tif');
ValPerimSablon2=961;
sablon3=imread('sablon3.tif');
ValPerimSablon3=705;
sablon4=imread('sablon4.tif');
ValPerimSablon4=659;
sablon5=imread('sablon5.tif');
ValPerimSablon5=621;
sablon6=imread('sablon6.tif');
ValPerimSablon6=566;
sablon7=imread('sablon7.tif');
ValPerimSablon7=531;
sablon8=imread('sablon8.tif');
ValPerimSablon8=449;
sablon9=imread('sablon9.tif');
ValPerimSablon9=387;
rulment=imread(RulDetectat);
figure,imshow(rulment);
title('Rulment Detectat');
aux=bwlabel(rulment);
stat1=regionprops(aux,'Perimeter');
ValPerimRulmen=floor(stat1.Perimeter);
if ValPerimRulmen>ValPerimSablon1-20&&ValPerimRulmen<ValPerimSablon1+40
    sablon=sablon1;
    TipRulment=1;
%     Diam=46.96;
%     h=20.55;
end
if ValPerimRulmen>ValPerimSablon2-20&&ValPerimRulmen<ValPerimSablon2+40
    sablon=sablon2;
    TipRulment=2;
%     Diam=46.96;
%     h=13.95;
end
if ValPerimRulmen>ValPerimSablon3-10&&ValPerimRulmen<ValPerimSablon3+40
    sablon=sablon3;
    TipRulment=3;
%     Diam=34.96;
%     h=9.9;
end
if ValPerimRulmen>ValPerimSablon4-2&&ValPerimRulmen<ValPerimSablon4+35
    sablon=sablon4;
    TipRulment=4;
%     Diam=31.96;
%     h=9.9;
end
```

```
if ValPerimRulmen>ValPerimSablon5-10&&ValPerimRulmen<ValPerimSablon5+35
    sablon=sablon5;
    TipRulment=5;
%     Diam=31.96;
%     h=7.9;
end
if ValPerimRulmen>ValPerimSablon6-0&&ValPerimRulmen<ValPerimSablon6+40
    sablon=sablon6;
    TipRulment=6;
%     Diam=27.96;
%     h=7.94;
end
if ValPerimRulmen>ValPerimSablon7-20&&ValPerimRulmen<ValPerimSablon7+30
    sablon=sablon7;
    TipRulment=7;
%     Diam=25.96;
%     h=7.94;
end
if ValPerimRulmen>ValPerimSablon8-20&&ValPerimRulmen<ValPerimSablon8+40
    sablon=sablon8;
    TipRulment=8;
%     Diam=21.96;
%     h=6.94;
end
if ValPerimRulmen>ValPerimSablon9-20&&ValPerimRulmen<ValPerimSablon9+40
    sablon=sablon9;
    TipRulment=9;
%     Diam=18.96;
%     h=5.94;
end
SablonUtilizat=sablon;
% TipulRulmentului=TipRulment
% DiametruExterior=Diam
% Inaltime=h
% figure,imshow(SablonUtilizat);
% title('Sablon');
aux1=bwlabel(SablonUtilizat);
% stat2=regionprops(aux1,'Perimeter');
% ValPerimSablon=stat2.Perimeter
stat3=regionprops(aux,'Centroid');%se determina centrul de greutate al
rulmentului din imagine
rulment_contur=bwperim(rulment);
rg=floor(stat3.Centroid(2)+.5);
cg=floor(stat3.Centroid(1)+.5);%pixelul  corespunzator  centrului  de
greutate al rulmentului
rulment_contur(rg,cg)=1;
% figure,imshow(rulment_contur);
% title('Contur Rulment Detectat cu CG');
[r_contur,c_contur]=find(aux1==1);
[rsg,csg]=find(aux1==2);
%deplasare sablon
dr=rg-rsg;
dc=cg-csg;
sablon_deplasat=zeros(size(sablon));
for i=1:length(r_contur)
```

```
    sablon_deplasat(r_contur(i)-dr,c_contur(i)-dc)=1;
end
```

Anexa 10: Programul din controlerul robotului pentru aplicaţia de simetrie

```
PROGRAM   SIMET

              *********************
   3633: SPEED     10
   3634: SPEEDL        5.000
   3635: SET       OUT[6] = 1
   3637: DELAY     50
   3638: SET       OUT[6] = 0
   3640: WAIT      IN[2] = 1
   3642: MOVE      CIM[150]
   3643: SETPVC    SIM[6] X  XS
   3644: SETPVC    SIM[6] Y  YS
   3645: SET       ROT=ROT - 74745
   3646: SETPVC    SIM[6] P  ROT
   3647: MOVED     SIM[5]
   3648: GOSUB     OGRIP
   3649: MOVELD    SIM[6]
   3650: GOSUB     CGRIP
   3651: MOVEL     SIM[5]
   3652: MOVE      CIM[150]
   3653: SET       LOC=10 + TIP
   3654: MOVED     SIM[LOC]
   3655: MOVELD    SIM[TIP]
   3656: GOSUB     OGRIP
   3657: MOVELD    SIM[LOC]
   3658: GOSUB     CGRIP
   3659: MOVED     CIM[150]
   3660: END
```

Anexa 11: Lista publicaţiilor personale publicate sub afiliere UPT

1. Lucrări ştiinţifice publicate în volumele unor manifestări ştiinţifice (Proceedings) indexate ISI Proceedings

1. **Pop Cristian**, Grigorescu Sanda Margareta, Davidescu Arjana –"Robot Vision Application for Bearings Identification and Sorting", Mechanisms, Mechanical Transmissions and Robotics, ISBN-13:978-3-03785-395-5, Applied Mechanics and Materials, vol. 162, ISSN 1660-9336 Trans Tech Publications, 2012, pp.523-531.

2. **Pop Cristian**, Grigorescu Sanda Margareta, Davidescu Arjana –"Colored object detection algorithm for visual-servoing application", Proc. 13th Int. Conf. on Optimization of Electrical and Electronic Equipment OPTIM 2012, Brasov, Romania, ISBN: 978-1-4673-1653-8/12, pp. 1539-1544, May 2012, ieee.

2. Lucrări ştiinţifice publicate în reviste de specialitate indexate BDI

1. Moldovan Florina, Dolga Valer, **Pop Cristian** "Kinetostatic Analysis of an articulated walking mechanism" , MECHANISMS AND MACHINE SCIENCE 3-MECHANISMS, TRANSMISSIONS AND APPLICATIONS, ISSN 2211-0984, ISBN 978-94-007-2726-7, Editura Springer Dordrecht Heilderberg London New York, 2012, pp. 103-111.

2. **Pop Cristian**, Davidescu Arjana, Moldovan Florina – "Object recognition using a smart camera", Romanian Review Precision Mechanics, Optics And Mechatronics, ISSN 1584-5982, Editura Cefin, Bucuresti, Romania, nr. 39, 2011, pp. 189-193.

3. Moldovan Florina, Dolga Valer, Ciontoş Ovidiu, **Pop Cristian** –"CAD Design and analytical model of a twelve bar walking mechanism", Scientific Bulletin. Series D: Mechanical Engineering, ISSN 1454-2358, Editura Politehnica Express, Bucureşti, România, vol. 73, nr.2, 2011, pp. 35-49.

4. Moldovan Florina, Dolga Valer, **Pop Cristian** -" Design and optimization method proposed for a new type of walking robot", Robotică şi Management-International Journal, ISSN 1453-2069, Robotics Society of Romania „Eftimie Murgu" University of Reşiţa Caraş-Severin County Council and „Politehnica" University of Timişoara,România vol. 15, No. 1, June 2010, pp. 41-47.

3. Lucrări ştiinţifice publicate în volumele unor manifestări ştiinţifice (Proceedings) indexate BDI

1. **Pop Cristian**, Davidescu Arjana –"Buildings Inspections by Image Processing Approach", Proceedings of the Fifth International Conference on Optimization of the Robots and Manipulators, OPTIROB 2010,Călimăneşti, România, ISBN 978-981-08-5840-7, May 2010, pp. 205-209.

4. Lucrări ştiinţifice publicate în volumele unor manifestări ştiinţifice

1. Pop Cristian – "Improvement of Automated Robot Assembly Process using Image Processing Methods", Workshop_ul nr.2 Interdisciplinaritatea şi managementul cercetării, Universitatea „Politehnica" din Timişoara, Secţiunea Mecanică, Timişoara pp. 45-46.