Interacțiunea grafică 2D în materiale educaționale

Andreea Chișcari
Universitatea Tehnică din Cluj-Napoca
Str. Memorandumului 28, Cluj-Napoca
andreeachiscari@gmail.com

Dorian Gorgan
Universitatea Tehnică din Cluj-Napoca
Str. Memorandumului 28, Cluj-Napoca
dorian.gorgan@cs.utcluj.ro

REZUMAT

Lucrarea de față prezintă principalele tehnici folosite în domeniul recunoașterii schițelor . Vom analiza problemele ridicate de construirea unui astfel de sistem .

Vom utiliza tehnicile studiate din domeniul recunoașterii schițelor pentru a crea un sistem de recunoaștere capabil să evalueze o schiță, având un model de referință pentru aceasta. Dorim să integrăm acest sistem într-o platformă eLearning pentru a-l utiliza în evaluarea răspunsurilor grafice ale studenților.

Sistemele de recunoaștere a schițelor s-au dovedit deja a fi unelte valoroase în educație. Domeniile în care sunt utilizate aceste sisteme sunt: circuite logice, circuite electrice, diagrame UML, mecanică, chimie, matematică etc.

Cuvinte cheie

Interacțiune om-calculator, platformă educațională.

Clasificare ACM

H5.2. Information interfaces and presentation (e.g., HCI): Miscellaneous.

INTRODUCERE

Oamenii folosesc schițele pentru a exprima și a stoca idei în mai multe domenii, precum: inginerie mecanica, design software, arhitectură, electronică, etc. Concepte care sunt dificil de explicat în cuvinte pot fi , de multe ori, explicate foarte ușor prin diagrame. Pe masură ce dispozitivele cu input bazat pe pen au devenit comune, s-au dezvoltat numeroase sisteme care se folosesc de acest tip de interacțiune.

Sisteme precum LADDER [3] s-au dovedit a fi capabile să recunoască schițe din mai mult de 30 de domenii, folosind tehnicile geometrice, generale.

Scopul nostru este acela de a prelua si îmbunatati tehnicile folosite de către sistemele de recunoaștere a schițelor existente pentru a crea propriul nostru sistem, capabil să ofere o evaluare calitativă schițelor. Acest sistem va fi utilizat în cadrul unei platforme eLearning pentru evaluarea răspunsurilor grafice ale studenților.

Deoarece corectarea raspunsurilor grafice necesită mult timp, acestea sunt adesea omise din examinările tradiționale și sunt înlocuite cu întrebari de tip grilă.

Scopul unei astfel de testări ar fi acela de a verifica dacă studentul percepe obiectele și a înțeles conceptele din spatele teoriei învațate.

Utilizarea unor întrebări care necesită răspuns grafic ajută profesorul prin faptul că oferă o măsură mai bună a progresului studenților decât o fac întrebările cu răspuns de tip grilă.

Metodele tradiționale de recunoaștere a schițelor pot fi grupate în trei categorii: bazate pe forme, bazate pe recunoașterea gesturilor și bazate pe relații geometrice. Deși fiecare din aceste metode are avantajele și dezavantajele sale, metodele geometrice s-au dovedit a fi cele mai potrivite pentru construirea unui sistem generic, care poate fi utilizat în mai multe domenii.

În recunoașterea folosind metode geometrice formele complexe sunt descrise prin combinații de forme mai simple între care există anumite relații geometrice sau constrângeri.

Problema este una de natură combinatorică. Pe masură ce numărul de forme desenate crește, timpul de calcul crește exponențial, ceea ce face ca sistemul să fie inutilizabil pentru probleme complexe, în domenii cu simboluri multe sau complicate. Cercetatorii au căutat diverse căi prin care procesul de recunoaștere poate fi accelerat: pruning, caching, indexare, structuri de date.

Unul dintre principiile care stau la baza unui astfel de sistem este acela că, în recunoașterea schițelor, ar trebui să li se permită utilizatorilor să interacționeze liber, natural. Dorim sa înlaturam, pe cât posibil, orice constrângere impusa utilizatorului la desenare.

Sistemul de recunoaștere trebuie să își susțina utilizatorul în timp ce acesta schițează, să vină cu informații relevante, dar să nu îl întrerupă pe acesta din procesul de desenare. Sistemul ideal de recunoaștere ar fi asemănător cu prezența unui observator uman care se uită peste umărul designerului și interpretează schița pe masură ce aceasta este desenată.

Secțiunea următoare analizează câteva realizări importante din acest domeniu, cu avantajele și dezavantajele fiecarei abordări. După această secțiune vom prezenta modul în care sistemul este integrat în contextul eLearning specificând pașii pe care profesorul și studentul trebuie să îi realizeze pentru a interacționa cu sistemul pentru a-și atinge obiectivele.

În continuare vom descrie modelul grafic folosit. Vom face o descriere a limbajului SKEL și a tipurilor de constrângeri definite de către acesta.Vom prezenta cațiva dintre algoritmii pe care i-am folosit și rezultatele obținute în urma aplicării acestora.

ALTE REALIZĂRI

Odata cu dezvoltarea tehnologiilor bazate pe pen, numărul tool-urilor/aplicațiilor care utilizează acest tip de interacțiune a crescut. Au aparut tool-uri în diverse domenii. Multe dintre acestea, însa, au ca obiectiv

îmbunatațirea experienței utilizator, crearea unei interfețe care sa îl ajute pe utilizator să realizeze anumite task-uri. Acestea nu sunt capabile, totuși, să ofere posibilitatea unei exprimări naturale, libere.

O încercare notabilă de a demonstra că recunoașterea schițelor se poate realiza folosind modele geometrice a fost sistemul Tahuti[5], un sistem pentru recunoașterea diagramelor UML. Tahuti combină libertatea de desenare cu puterea de procesare pusă la dispozitie de catre o diagramă interpretată. Sistemul este construit din mai multe nivele și recunoaște obiecte folosindu-se de proprietațile geometrice ale acestora.

Avantajul sistemului Tahuti constă în faptul că domeniul diagramelor UML nu este unul complex. Obiectivul, însa este acela de a se ajunge la o soluție generală, care sa poată fi aplicată cu succes în mai multe domenii.

Sistemul SketchRead[1] dezvoltat de catre Alvorado și Davis este un sistem multi-domain capabil să recunoască diagrame din mai multe domenii. Totuși, acuratețea acestuia pentru schițe complicate este înca mica pentru a putea fi folosit în practică. Totodată, timpul de procesare crește foarte mult pe măsură ce numărul de forme desenate crește. Ocazional, sistemul ajunge să ruleze pentru perioade foarte lungi.

În dorința de a descoperi o soluție de reducere a spațiului de căutare, numeroase sisteme au încercat să impuna o serie de constrângeri. Anumite sisteme presupun că desenarea unui obiect se realizează folosind gesturi consecutive. Alte sisteme limiteaza spațiul de cautare printr-o abordare greedy - odata ce o formă a primit o interpretare, toate componentele sale nu mai sunt luate în considerare pentru interpretarile viitoare.

Alte sisteme se bazează pe feedback din partea utilizatorului pentru a menține corectitudinea interpretărilor[6]. Astfel, o schiță este recunoscută în timp ce este desenată. Când sistemul realizează o interpretare greșită a schiței, utilizatorul are posibilitatea de a corecta imediat acest lucru.

Foarte multe din aceste sisteme utilizează constrângeri spațiale pentru a limita spațiul de căutare. Acestea consideră pentru analiză numai grupuri de linii/curbe care sunt într-o anumită zona definită.

INTERACTIUNEA GRAFICA IN CONTEXTUL ELEARNING

Pentru a utiliza întrebări care necesită răspuns grafic și pentru ca sistemul să poată corecta/evalua automat răspunsul unui student și să îi ofere acestuia feedback imediat, profesorul trebuie ca mai întâi să ofere o soluție de referință pentru fiecare problemă, trebuie să descrie răspunsul așteptat. Pentru aceasta se utilizează un editor grafic în cadrul căruia profesorul specifică prin desen răspunsul. Sistemul folosește baza sa de cunoștințe, formată din forme primitive și constrângeri) pentru a recunoaște primitivele grafice și relațiile dintre acestea și transformă schița profesorului în limbaj SKEL.

Evident, sistemul suportă și recunoaște multe constrângeri, multe din acestea nefiind relevante/importante pentru evaluare, de aceea considerăm a fi necesar un pas în care profesorul să aibă posibilitatea să modifice descrierea SKEL creată de către sistem (Figura 1).

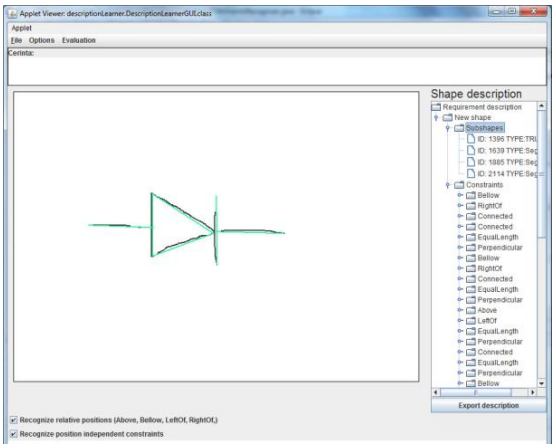

Figura 1. Tool utilizat de catre profesor pentru specificarea raspunsului de referinta

De asemenea, profesorul trebuie să aleagă modul în care dorește ca sistemul să interacționeze cu studentul, dacă dorește ca sistemul să ii ofere feedback studentului sau nu.

Studentul va avea la dispoziție un panou grafic în care va specifica răspunsul pe care îl consideră corect. În modul de lucru cu feedback activat el va confirma sau infirma fiecare ipoteză, având posibilitatea să își modifice răspunsul (Figura 2).

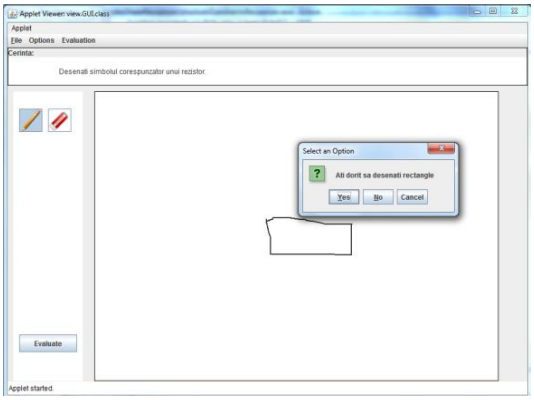

Figura 2. Panou utilizat de catre student pentru specificarea raspunsului grafic. Mod de lucru cu feedback activat.

La finalul fiecarui ciclu de evaluare, sistemul genereaza un fișier xml în care descrie formele principale pe care le-a recunoscut și acuratețea cu care acestea au fost desenate.

Pentru a oferi o măsură a calității răspunsului grafic al studentului, sistemul se folosește de o serie de parametri care vor fi definiți pentru fiecare tip de constrângere. Un exemplu de astfel de parametru este distanța maximă acceptată dintre două forme pentru ca acestea să poata fi înca considerate conexe .

Sistemul trebuie să poată funcționa atât în timp real, oferind feedback studentului în timp ce acesta specifică

răspunsul grafic, cât și la o cerere a profesorului. Vom lăsa la latitudinea profesorului selectarea modului în care dorește să se realizeze evaluarea, cu sau fără feedback pentru student. În cazul în care feedbackul este activat, sistemul trebuie să fie capabil să se folosească de acesta.

MODELAREA COMUNICĂRII GRAFICE

Modelul conceptual

Se consideră că o schiță este alcatuită dintr-o colecție de forme, pentru care sistemul are definiții, între care există relații geometrice.

Numim forme primitive următoarele: punct, segment, arc, triunghi, dreptunghi, romb, pătrat, paralelogram etc. Considerăm trei forme primitive de bază: punctul, segmentul și arcul. Toate celelalte forme pot fi considerate ca fiind alcătuite din cele trei forme de bază.

Deși le numim tot primitive, triunghiul, pătratul etc. sunt forme compuse și există o descriere pentru acestea în limbaj SKEL. Diferența dintre formele primitive compuse și formele complexe este faptul că formele primitive compuse sunt alcătuite numai din cele trei tipuri primitive de bază: punct, segment, arc, cerc, elipsă.

Am considerat că acest set de forme primitive este suficient pentru construirea unor forme complexe din mai multe domenii. Experimentele realizate de către cei de la Massachusetts Institute of Technology au demonstrat aplicabilitatea acestei metode, folosind aceleași primitive, în peste 30 de domenii diferite.

Deoarece descrierile SKEL sunt ierarhice, orice formă nou definită poate fi utilizată ca formă componentă în descrierea unor forme de nivel mai înalt.

Constrângerile suportate de către sistem sunt cele definite de limbajul SKEL, descrise în secțiunea următoare.

Limbajul SKEL

Limbajul SKEL este un limbaj grafic în format XML, derivat din limbajul LADDER[3,4]. Acesta oferă o descriere structurală, ierarhică a unei schițe, folosind alte forme, definite anterior, și relațiile geometrice dintre acestea.

Definiția unei forme constă în specificarea celor două secțiuni: Subshapes și Constraints.

Secțiunea Subshapes constă în enumerarea unui set de forme care au o etichetă asociată și un tip. Tipul unei forme reprezintă numele unei forme definite anterior sau al unei forme primitive. Eticheta este utilizată pentru a putea face referințe ulterioarea la aceste forme componente. Acest set de forme va intra în compoziția noii forme care se definește.

În figura 3. este prezentat un exemplu de descriere în limbaj SKEL a simbolului grafic corespunzător unei diode din domeniul circuitelor electrice. În acest exemplu, în secțiunea Subshapes, se specifică faptul că noua formă care se definește este alcătuită dintr-un triunghi și trei segmente.

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<Shape name="Dioda">
    <Subshapes>
        <Subshape id="a" type="TRIANGLE"></Subshape>
        <Subshape id="b" type="segment"></Subshape>
        <Subshape id="c" type="segment"></Subshape>
        <Subshape id="d" type="segment"></Subshape>
    </Subshapes>
    <Constraints>
        <Perpendicular id="perpendicular 1">
            <Segment>c</Segment>
            <Segment>d</Segment>
        </Perpendicular>
        <Perpendicular id="perpendicular 2">
            <Segment>b</Segment>
            <Segment>a.b</Segment>
        </Perpendicular>
        <Parallel>
            <Segment>c</Segment>
            <Segment>a.b</Segment>
        </Parallel>
    </Constraints>
</Shape>
```

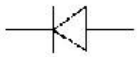

Figura 3. Descriere în limbaj SKEL a simbolului corespunzător unei diode din domeniul circuitelor electrice

Fiecare dintre formele care compun o formă complexă trebuie să fie etichetate, să primească un id. Pentru a referi o subcomponentă a unei forme compuse se folosește operatorul '.'. De exemplu, notația din figura 3, 'a.p1' referă un capat al segmentului a.

Limbajul SKEL definește un set de constrângeri care reprezintă defapt posibilele relații geometrice dintre componentele unei forme complexe. Noile forme sunt definite utilizând forme care au fost definite anterior și relațiile geometrice dintre acestea.

Câteva dintre constrângerile definite de limbajul SKEL, câteva preluate din limbajul LADDER, sunt exemplificate în Figura 4.

```
//Forma a are centrul deasupra centrului formei b
    <Above>
        <Shape>a</Shape>
        <Shape>b</Shape>
    </Above>
//Forma a are centrul la dreapta centrului formei b
    <RightOf>
        <Shape>a</Shape>
        <Shape>b</Shape>
    </RightOf>
//Segmentele s1 si s2 sunt perpendiculare
    <Perpendicular>
        <Segment>s1</Segment>
        <Segment>s2</Segment>
    </Perpendicular>
//Segmentele s1 si s2 sunt paralele
    <Parallel>
        <Segment>s1</Segment>
        <Segment>s2</Segment>
    </Parallel>
//Formele a si b sunt conectate
    <Connected>
        <Shape>a</Shape>
        <Shape>b</Shape>
    </Connected>
//Punctele p1 si p2 coincid
    <Equals>
        <Element>p1</Element>
        <Element>p2</Element>
    </Equals>
//Segmentele s1 si s2 au aceeasi lungime
    <EqualLength>
        <Segment>s1</Segment>
        <Segment>s2</Segment>
    </EqualLength>
```

Figura 4. Constrângeri definite de limbajul SKEL

Constrângerile definite sunt de mai multe tipuri. Există constrângeri care sunt independente de poziția și orientarea schiței în scenă. Astfel de constrângeri sunt: Coincident, Connected, Intersects, Included, Angle, Equals, Parallel, Perpendicular etc. De asemenea, limbajul SKEL definește și un set de constrângeri care sunt dependente de orientare, de poziția schiței în scenă. Astfel de constrângeri sunt: Horizontal, Vertical, Above, Bellow, LeftOf, RightOf etc. Unele din aceste constrângeri pe care le-am definit pot fi descrise utilizând alte constrângeri. De exemplu, relalția RightOf poate fi descrisă utilizând LeftOf cu inversarea ordinii parametrilor. Am decis, totuși, să introducem aceste constrângeri pentru a simplifica descrierile și a le face mai ușor de citit.

Pe lângă aceste tipuri de constrângeri geometrice, am decis introducerea în limbajul SKEL a unui set de constrângeri logice, cu scopul de a ușura procesul de descriere. Acestea oferă support în descrierea natural, logică a unei figure. De exemplu, pentru a descrie faptul că două drepte nu sunt paralele, fără a avea la dispoziție aceste funcții logice, ar trebui să căutăm o modalitate prin care să specificăm faptul că unghiul dintre acestea este diferit de 180º. Acest tip de descriere este, însă, mai puțin naturală. Ar fi mai ușoară o specificare care utilizează constrângerea Parallel definită anterior, însoțită de funcția logică NOT. În figura 5 este prezentată descrierea relației NOT(Parallel) în limbaj SKEL.

```
<Not>
    <Parallel>
        <Segment>a</Segment>
        <Segment>b</Segment>
    </Parallel>
</Not>
```

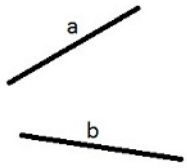

Figura 5. Exemplu de constrangere de tip logic

PRELUCRAREA ȘI RECUNOAȘTEREA SCHIȚEI

Dorința noastră este de a utiliza tehnicile studiate din domeniul recunoașterii schițelor pentru a crea un sistem de recunoaștere capabil să evalueze o schiță, având un model de referință pentru aceasta. Dorim să integrăm acest sistem într-o platformă eLearning pentru a-l utiliza în evaluarea răspunsurilor grafice ale studenților. Vom analiza, în continuare, principalele probleme pe care le ridică construirea unui astfel de sistem.

Sistemul este constituit din mai multe nivele. Am identificat patru pași principali pe care sistemul îi va efectua pentru recunoașterea formelor (Figura 6).

Primul pas efectuat de către sistemul nostru este acela de a reeșantiona inputul primit de la utilizator. Acest input este eșantionat la o rată fixă, determinată de către sistemul hardware și software utilizat. Prin urmare, viteza de desenare va avea efect asupra numărului de puncte dintr-un gest. După pasul de reeșantionare, gestul utilizator va fi constituit dintr-un set de puncte echidistante. Acest pas ajută atât la reducerea zgomotului din intrare cât și la creșterea performanțelor următorului pas de prelucrare, detecția colturilor.

Cel de-al doilea pas constă în detectarea formelor primitive de nivel inferior din inputul primit de la utilizator. Inputul reeșantionat este trimis către modulul de recunoaștere a formelor primitive. Acest modul încearcă să potrivească setul de puncte primit ca input la setul de primitive pentru care există itemi de recunoaștere: LineRecognizer, ArcRecognizer, PolylineRecognizer, CircleRecognizer și EllipseRecognizer.

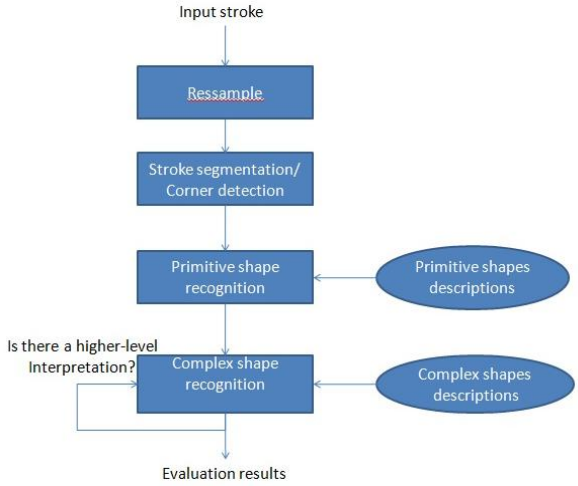

Figura 6. Niveluri de prelucrare

Pentru calcularea parametrului error itemii de recunoaștere se vor folosi de o trăsătură propusă de un grup de cercetători de la Universitatea Nanjing din China [8]. Aceștia au venit cu ideea utilizării unei trăsături de tip arie, care, deși nu este capabilă sa ofere informație pentru a distinge între două forme, este mult mai rezistentă la zgomot decât celelalte tipuri de trăsături. În modulul Recognizers, această trăsătură este utilizată numai pentru ghidarea recunoașterii și pentru verificarea rezultatelor.

Recunoașterea liniei

Pentru a determina dacă un gest este sau nu linie, trebuie să se verifice mai multe condiții.

Se calculează distanța ortogonală dintre cea mai bună linie și punctele corespunzătoare gestului. Această distanță este împărțită la lungimea totală a gestului pentru a determina un coeficient de eroare, care trebuie să fie mai mic decât un prag fixat.

De asemenea, se calculează și trăsătura linie pentru acest segment utilizând tehnica prezentată anterior. Această valoare trebuie să fie și ea mai mică decât un anumit prag fixat.

Recunoașterea arcului de cerc

Pentru acest item de recunoaștere am considerat arcele ca fiind segmente dintr-un cerc. Prin urmare, pentru a determina care este cel mai potrivit arc pentru setul de puncte primit ca parametru trebuie să determinăm cel mai potrivit cerc din care acest arc face parte. Pentru aceasta, se calculează centrul ideal al acestui cerc. Apoi se calculează o rază ideală pentru cerc, folosind media distanțelor dintre punctele gestului și centrul calculat.

În final se calculează trăsătura arie pentru arc si se verifică dacă aceasta are valoarea mai mică decât un prag stabilit.

Recunoaşterea elipsei

Modulul responsabil de recunoaşterea elipselor este capabil de a recunoaşte toate tipurile de elipse, fie acestea rotite sau nu.

Pentru recunoaşterea unei elipse se parcurg următorii paşi:

1. Se determină axa majoră prin cautarea a două puncte care se află la distanţa cea mai mare.
2. Se determină centrul elipsei prin calcularea mediilor pe x şi y a celor două puncte.
3. Axa minoră se construieşte ducând perpendiculara pe axa majoră în centrul acesteia. Se calculează apoi punctele de intersecţie ale acesteia cu punctele din gestul primit.

Pentru ca modulul să recunoască o elipsă trebuie să verifice cel puţin două condiţii. Prima se referă la testul de formă închisă, adică se va verifica dacă punctul de început a gestului coincide sau este apropiat de punctul de sfârşit. De asemenea, trăsătura arie trebuie să fie mai mică decât o valoare fixă, stabilită.

Recunoasterea cercului

Pentru recunoaşterea cercului se vor folosi multe din testele realizate de către itemul de recunoaştere a elipsei. Se va calcula, mai întâi, o rază ideală, ca fiind media distanţei dintre fiecare punct al gestului şi centrul ideal calculat în testul pentru elipsă.

Pentru a verifica faptul că un gest se potriveşte mai bine unui cerc decât unei elipse se va calcula raportul dintre axa majoră si axa minoră. De asemenea se efectuează şi testul care se foloseşte de trăsătura arie.

Recunoaşterea poliliniei

Testul de polilinie începe prin segmentarea gestului primit într-un set de sub-gesturi. Pentru acest lucru se utilizează algoritmul de recunoaştere a colţurilor din pachetul CornerDetection. Fiecare sub-gest este trimis apoi către itemii de recunoaştere a liniilor şi arcelor. Pentru fiecare astfel de sub-gest se însumează erorile parţiale care sunt normalizate prin împărtirea la lungimea totală a gestului. Pentru ca un gest să poată fi considerat polilinie trebuie ca toate sub-gesturile sale să fie clasificate ca segmente sau arce.

Recunoaşterea formelor complexe

În urmatoarea etapă sistemul utilizează descrierile formelor în limbaj SKEL pentru a recunoaşte forme complexe. Dorim ca interpretarea formelor complexe să funcţioneze recursiv, până la descoperirea interpretării de la nivelul cel mai înalt posibil.

Pentru a putea funcţiona în timp real şi pentru a putea oferi feedback, la fiecare gest schiţat de către utilizator, sistemul încearca să combine inputul nou cu alte forme desenate anterior pentru a vedea dacă se pot forma/ recunoaşte noi forme.

Nu este necesar să se verifice toate combinările din scenă. Se vor genera numai combinările care conţin informaţia proaspăt desenată, deoarece restul formelor desenate anterior au fost evaluate deja. Pentru fiecare linie (curbă sau polilinie) desenată de către utilizator pot fi identificate trei posibilitaţi de interpretare. Aceasta poate să fie o formă primitivă de sine stătătoare, poate face parte dintr-o formă complexă, sau poate să nu primească nici o interpretare, rămânând la interpretarea sa iniţială, de linie/curbă sau polilinie.

Putem aplica tehnici de grupare pentru recunoaşterea formelor primitive. Formele primitive sunt descrise şi ele în limbajul LADDER. Diferenţa dintre aceste forme şi alte forme complexe este faptul că am definit aceste forme ca fiind conexe. Din acest motiv se va aplica algoritmul de recunoaştere numai pe grupuri conexe, fapt care duce la o creştere considerabilă a performanţei .

Tot pentru creşterea eficienţei, multe dintre constrângerile suportate de către sistem sunt identificate în timpul desenării. Astfel, în momentul evaluării se vor genera doar ipotezele pentru care s-au recunoscut atât formele componente cât şi constrângerile corespunzătoare.

Sistemul nostru ar putea beneficia şi de feedback-ul din partea utilizatorului în legatură cu ceea ce acesta intenţionează să deseneze. În cazul unei interpretări greşite, utilizatorul are posibilitatea să işi modifice schiţa. Pentru aceasta sistemul trebuie să pună la dispoziţia utilizatorilor o funcţie de ştergere. Utilizatorul are posibilitatea să şteargă numai porţiuni din desenul său. Dupa ştergere, zona în care s-au făcut modificările ar trebui reevaluată.

EXPERIMENTE

Segmentarea

Pentru a determina parametrii optimi de intrare necesari algoritmului de recunoaştere a punctelor de curbură maximă a unei linii curbe am analizat rezultatele obţinute de către autorii algoritmului [7] şi am rulat o serie de teste proprii.

Rezultatele obţinute au fost asemănătoare cu cele obinute la [7]. Toate unghiurile mai mici decât alpha=160 de grade vor fi considerate puncte locale de curbură , într-o vecinătate cuprinsă între 5 şi 7 puncte vecine.

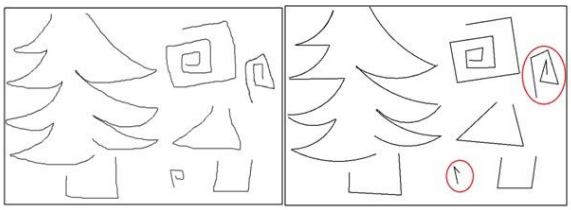

Figura 7. Test de segmentare

De asemenea, am observat faptul că rata de reeşantionare are un efect destul de important asupra rezultatului segmentării. Acest lucru se datorează faptului că o distanţă prea mare între punctele inputului reşeesantionat duce la pierderea detaliilor. O rată prea mică, poate la amplificarea zgomotului.

Experimental am dedus o rată optimă de 5 pixeli. Însă, pentru această valoare s-au observat erori ale algoritmului în recunoaşterea formelor de dimensiuni mici. În figura 7 sunt încercuite câteva erori ale sistemului datorate acestei probleme.

Recunoașterea formelor primitive de bază

Pentru recunoașterea primitivelor de bază și a primitivelor de nivel superior am rulat un set de teste. Pentru fiecare tip de primitivă de bază am construit o mulțime de 20 de imagini de testare pe care noi le considerăm a fi corecte. Pe lângă acestea, am încercat să forțăm sistemul de recunoaștere printr-un set de imagini despre care se știe că sunt incorecte, dar se consideră că pot să ridice probleme.

În tabelul din figura 8 sunt prezentate rezultatele acestor teste.

Forma	Rezultate test pozitiv (Numărul de exemple clasificate corect)	Rezultate test negativ (Numărul de exemple clasificate corect)
Linie	20/20(100%)	19/20(95%)
Arc de cerc	15/20(75%)	19/20(95%)
Cerc	19/20(95%)	20/20(100%)

Figura 8. Rezultatele testului de evaluare pentru recunoașterea primitivelor simple

Recunoașterea formelor primitive compuse

În tabelul din figura 9 sunt prezentate rezultatele evaluării sistemului de recunoaștere pentru forme primitive compuse. Pentru aceste teste am fixat parametrii modulului de segmentare cu valorile optime obținute la testul de segmentare: dmin = 3, dmax = 5, alpha = 150º. Pentru modulul de reeșantionare am folosit o rată egală cu 5 pixeli.

Forma	Rezultate test pozitiv (Numărul de exemple clasificate corect)	Rezultate test negativ (Numărul de exemple clasificate corect)
Triunghi	20/20(100%)	19/20(95%)
Dreptunghi	18/20(90%)	20/20(100%)
Pătrat	17/20(85%)	20/20(100%)
Paralelogram	17/20(85%)	18/20(90%)

Figura 9. Rezultatele testului de evaluare pentru recunoașterea formelor primitive compuse

Trebuie precizat faptul că recunoașterea acestor primitive compuse este dependentă de parametrii definiți pentru fiecare tip de constrângere utilizată în descriere. Valorile utilizate la rularea acestui test sunt prezentate în figura 10.

```
# The maximum accepted error (in degrees) for the Angle constraint
AngleConstraint.error=10.0

# The maximum accepted error (in pixels) for the EqualLength constraint
EqualLengthConstraint.error=20.0

# The maximum accepted error (in pixels) for the Equals constraint
EqualsConstraint.error=15.0

# The maximum accepted error (in degrees) for the Perpendicular constraint
PerpendicularConstraint.error=10.0

# The maximum accepted error for the Parallel constraint
ParallelConstraint.error=20.0

# The maximum accepted error (in pixels) for the Connected constraint
ConnectedConstraint.error=20.0

# The maximum accepted error (in degrees) for one angle in the sumAngles constraint
SumAnglesConstraint.error=20.0
```

Figura 10. Parametrii utilizați pentru fiecare tip de constrângere

Test de stres

Deoarece sistemul ar trebui să poată să funcționeze online, să facă recunoașteri în timp real, am decis să realizam o serie de teste de stres (Figura 10). Pentru recunoașterea formelor primitive, sistemul a suportat până la 80 de gesturi de input, și a recunoscut corect primitive, având un timp de răspuns mai mic decât 1.5-2 secunde.

Figura 10. Test de stres

CONCLUZII

Sistemele de recunoaștere a schițelor care folosesc metode geometrice au avantajul de a fi generice, putând fi aplicate cu succes în numeroase domenii. Există totuși domenii în care acest tip de sisteme nu pot fi utilizate. Un sistem care se folosește de un limbaj bazat pe metode geometrice nu ar putea fi aplicat pentru descrierea formelor artistice, cum ar fi un portret. Sunt suportate numai formele complexe care pot fi descrise prin formele primitive definite de către sistem. De asemenea, sistemele de acest tip nu s-au dovedit a fi foarte eficiente în domenii cu forme ce conțin multe curbe, deoarece acestea sunt dificil de descris în detaliu.

REFERINȚE

1. C. Alvarado. *Multi-Domain Sketch Understanding*. PhD thesis, Massachusetts Institute of Technology, 2004.
2. Alvarado C.: A Natural Sketching Environment: Bringing the Computer into Early Stages of Mechanical Design. Master's thesis, MIT, 2000.
3. Hammond T.: Ladder: A perceptually-based language to simplify sketch recognition user interface development. PhD Thesis, Massachusetts Institute of Technology, 2007.
4. Tracy Hammond and Randall Davis. *Ladder: A language to describe drawing, display, and editing in sketch recognition*. Proceedings of the 2003 International Joint Conference on Artificial Intelligence (IJCAI), 2003.
5. Hammond T., Davis R.: *Tahuti: A geometrical sketch recognition system for UML class diagrams.* AAAI Spring Symposium on Sketch Understanding (March 25-27), 59–68, 2002.
6. Jason Hong, James Landay, A. Chris Long, and Jennifer Mankoff. *Sketch recognizers from the end-user's, the designer's, and the programmer's perspective.* Sketch Understanding, Papers from the 2002 AAAI Spring Symposium, pages 73–77, March 25-27, 2002.
7. Dmitry Chetverikov , Zsolt Szabo. *A simple and efficient algorithm for detection of high curvature points in planar curves.* Image and Pattern Analysis Group. Computer and Automation Research Institute, Budapest, 1999.
8. Yu, B. and Cai, S. *A Domain-Independent System for Sketch Recognition.* In Proc. of the 1st International Conference on Computer Graphics and Interactive Techniques in Australasia and South East Asia, ACM Press (2003),141-146